NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

OPERATIONAL IMPLEMENTATION OF ERS SATELLITE SCATTEROMETER WIND RETRIEVAL AND AMBIGUITY REMOVAL

by

Christy A. Martin

December 1996

Thesis Advisors: Marie Colton, Sandra Scrivener

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE December 1996	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE OPERATIONAL IMPLEMENTATION OF ERS SATELLITE SCATTEROMETER WIND RETRIEVAL AND AMBIGUITY REMOVAL			5. FUNDING NUMBERS
6. AUTHOR(S) Martin, Christy A.			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Fleet Numerical Meteorology and Oceanography Center, Monterey, California			10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U. S. Government.			
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

The European Remote Sensing (ERS) satellite's wind scatterometer is a microwave radar which provides the only source of global data for both wind speed *and* direction. The European Space Agency uses this data to generate a wind Fast Delivery Product (FDP). However, this product is insufficient in its resolution of the scatterometer's inherent wind direction ambiguity. This thesis presents development of improved processing of ERS satellite wind scatterometer data at the Fleet Numerical Meteorology and Oceanography Center (FNMOC) for dissemination to Navy operational centers. Discussion includes an introduction to the physical principles and operation of the scatterometer instrument, past and current systems, and development of the model transfer function. An alternative method of producing the wind field from raw scatterometer data is presented. This processing method for raw scatterometer data was developed for FNMOC, using the local global model (NOGAPS) wind field for comparison. The resulting scatterometer wind field consistently provides a more realistic wind field than the FDP, as demonstrated in the specific example of hurricane Hortense in the Caribbean Sea on 12 September 1996. Further comparison with the NOGAPS wind field, the Defense Meteorological Satellite Program Special Sensor Microwave/Imager wind speeds, and in-situ measurements provide additional validation.

14. SUBJECT TERMS European Remote Sensing Szatellite, Scatterometer, Fleet Numerical Meteorology and Oceanography Center			15. NUMBER OF PAGES 146
			16. PRICE CODE
17. SECURITY CLASSIFI-CATION OF REPORT Unclassified	18. SECURITY CLASSIFI-CATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFI CATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)

Prescribed by ANSI Std. 239-18 298-102

OPERATIONAL IMPLEMENTATION OF ERS SATELLITE SCATTEROMETER WIND RETRIEVAL AND AMBIGUITY REMOVAL

Christy A. Martin
Lieutenant (junior grade), United States Navy
B.S., United States Naval Academy, 1993

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ASTRONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1996

Author: Christy A. Martin
Christy A. Martin

Approved by: Marie Colton
M. Colton, Thesis Advisor

Sandra L. Scrivener
S. Scrivener, Thesis Advisor

Daniel J. Collins
Daniel J. Collins, Chairman
Department of Aeronautical and Astronautical Engineering

ABSTRACT

The European Remote Sensing (ERS) satellite's wind scatterometer is a microwave radar which provides the only source of global data for both wind speed *and* direction. The European Space Agency uses this data to generate a wind Fast Delivery Product (FDP). However, this product is insufficient in its resolution of the scatterometer's inherent wind direction ambiguity. This thesis presents development of improved processing of ERS satellite wind scatterometer data at the Fleet Numerical Meteorology and Oceanography Center (FNMOC) for dissemination to Navy operational centers. Discussion includes an introduction to the physical principles and operation of the scatterometer instrument, past and current systems, and development of the model transfer function. An alternative method of producing the wind field from raw scatterometer data is presented. This processing method for raw scatterometer data was developed for FNMOC, using the local global model (NOGAPS) wind field for comparison. The resulting scatterometer wind field consistently provides a more realistic wind field than the FDP, as demonstrated in the specific example of hurricane Hortense in the Caribbean Sea on 12 September 1996. Further comparison with the NOGAPS wind field, the Defense Meteorological Satellite Program Special Sensor Microwave/Imager wind speeds, and in-situ measurements provide additional validation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to acknowledge the support of Fleet Numerical Meteorology and Oceanography Center, Monterey, California, for providing the resources used for this thesis. I would also like to extend my appreciation to Prof. Sandra Scrivener and Dr. Marie Colton for their guidance and patience during the work in completing this thesis.

I. INTRODUCTION

A. NAVAL RELEVANCE: APPLICATION OF SEA SURFACE WIND INFORMATION

The success of earth observation from space in the last 30 years has provided a dramatic new way to observe and study the earth as a natural system. The implications are many, but perhaps none more profound than the effect of global information on oceanography and meteorology. Accordingly, oceanography from space has enjoyed nearly full-time utility, in either image or data formats, since the launch of Seasat in 1978, the first satellite specifically designed for ocean observation. Prior to the advent of satellite remote sensing, sea surface information was provided only by land stations, buoys, ships, and aircraft. Together these mechanisms provided an extremely limited and incongruous picture of the state of the sea surface.

In naval operations, knowledge of the sea state is a fundamental parameter in the strategic and tactical equation. Sea surface wind is the primary forcing mechanism of sea state and, therefore, is a significant factor in the operational decision making process. The accuracy with which surface wind can be predicted directly influences the accuracy with which the strategic or tactical outcome can be predicted. The wind scatterometer instrument on board the European Space Agency's (ESA) remote sensing satellite is a space-based radar that provides both wind speed and direction. The information available from the ESA's wind scatterometer is expected and has been demonstrated (to a limited degree) to have a direct positive impact on global and regional forecasts as well as the prediction and analysis of extreme weather phenomena, such as tropical cyclone and hurricane development, high winds and seas, and storm frontal structure. Wind *speed* information has been available for some time from the Defense Meteorological Satellite Program

(DMSP) Special Sensor Microwave/Imager (SSM/I) passive microwave radiometer, but only the scatterometer provides a measure of wind *direction* from space.

Agencies such as the Joint Typhoon Warning Center (JTWC) in Guam and the Tropical Prediction Center/National Hurricane Center (TPC) in Miami, Florida, provide short-term forecasts and warnings for extreme weather conditions over their respective areas of operation to the U.S. Navy fleet. Additionally, the U.S. Navy's Fleet Numerical Meteorology and Oceanography Center (FNMOC) provides global and regional forcasts and data to the fleet. These centers rely on satellite data to perform forecasting and warning duties. Primarily geosynchronous satellite visual imagery and the SSM/I have provided the means to formulate an understanding of the surface wind field. However, geosynchronous visual imagery of surface features is limited by coarse resolution at hight latitude, oblique viewing angles, high-level clouds, and other meteorological phenomena. The SSM/I is sensitive to rain contamination, which often occurs in high wind areas of prime interest. In this respect, the scatterometer wind vectors directly complement the SSM/I data. The JTWC has used the wind scatterometer data in operational situations and has found substantial improvements in the ability to accurately analyze the current surface wind field, even with lower quality scatterometer data which has some residual directional error.

Sea surface wind is one of the most critical factors in many facets of naval operations. Not only does surface wind drive other meteorological phenomena such as sea state, but directly affects the decisions made in both strategic and tactical operational situations from the execution of an amphibious assault to the launching of carrier aircraft. The more accurately the wind condition can be analyzed and predicted, the less costly military operations will be in equipment, time, and lives.

B. OPERATIONAL PROCESSING OF SCATTEROMETER WINDS

A radar scatterometer is the name given to an oblique-viewing active microwave radar, measuring the returned (backscattered) radar energy from an area illuminated on the sea-surface at a particular frequency. The amplitude of the backscattered signal is interpreted empirically as a measure of the ocean surface roughness, which can then be related to the surface wind speed and its related stress [Ref. 1]. In the late 1960's, the first microwave radar with the purpose of measuring sea surface winds was developed to demonstrate that such an instrument on a space-based platform could obtain global ocean wind data. To date, scatterometers have been flown on aircraft, Skylab [Ref. 2], Seasat [Ref. 1], and ERS-1 and 2 [Ref. 3]. These successful flights have contributed to increased understanding of the relationship between radar measurements and ocean wind vectors. However, incomplete specification of the physics and limitations of empirical models result in errors in the scatterometer-derived wind vectors that must be corrected to meet operational requirements.

Currently there are two operating satellites with scatterometers as part of their payload: the European Remote Sensing Satellite (ERS-2), developed and operated by the European Space Agency (ESA), and the Japanese Advanced Earth Observing Satellite (ADEOS), carrying the NASA-Scatterometer (NSCAT). The ERS-2 succeeded the ERS-1 and was launched in April 1995. The ERS-2 has only been supplying valid scatterometer data since March 1996, after correction of a post-launch power supply problem, and took data collection duties over from ERS-1 on 3 June 1996. From ERS scatterometer data, ESA produces Fast Delivery Products which are designed to be available to the customer within three hours of observation. ADEOS was launched on 16 August 1996 and has not yet begun to supply operational data.

The National Center for Environmental Prediction (NCEP) began receiving the Fast Delivery Product (FDP) scatterometer wind data in early 1992. A preliminary study of the data revealed deficiencies in the FDP wind vectors, primarily involving the direction, such that the wind vectors were unacceptable for use in analysis and forecast models. [Ref. 4] Because the raw measurement data accompanies the FDP wind vectors, NCEP was able to develop a procedure to process the data locally. In preparation for this effort, NCEP performed a study on potential algorithms for wind retrieval. Scatterometer data, buoy and model data were collected over a period of one year, from 9 September 1993 to 9 September 1994. The scatterometer data were processed using five different algorithms (developed by various meteorological and educational institutions) and compared to the FDP and buoy and model data. The model function known as CMOD4 was determined to provide the most accurate wind retrieval and was chosen by NCEP for operational scatterometer processing.

NCEP processes the scatterometer data with the goal of supplying it to their local analysis and forecast models. Many other meteorological centers around the world are using the FDP for the same purpose, such as the French Institut Francais de Recherche pour l'Exploitation de la MER (IFREMER), the European Center for Medium-Range Weather Forecasts, the United Kingdom Meteorological Office, the Norwegian Meteorological Institute, and NASA's Goddard Space Flight Center. The most intensive studies of the impact of scatterometer data on numerical modeling show substantial modifications to global surface winds and generally an improved forecast, primarily in the Southern Hemisphere where there is traditionally a lack of wind observations. Other centers are using the data for purposes ranging from high wind and tropical storm warnings to numerical wave modeling.

The Fast Delivery Product has also been used in the generation of wind barb graphics by both the Naval Oceanographic Office (NAVO) and the National Environmental Satellite, Data, and Information Service (NESDIS). Initially, because of the known problems in the FDP, questionable vectors were manually removed before NAVO generated its vector graphic. However, increased demand and complexity have rendered this method impractical. NAVO has looked to the World Wide Web product produced by NESDIS as an example of how Navy users might access the scatterometer data in near real-time. The plotted vectors are only updated approximately four times daily, however, and are not suitable for real-time use. There is no established method for operational Navy users to supply feedback for the product's generation and questions of liability arise if the products are to be used in an operational situation. NAVO has, therefore, begun producing a similar graphic for dissemination over the World Wide Web. This process, however, places the responsibility of removing questionable vectors on the user.

The Fleet Numerical Meteorology and Oceanography Center (FNMOC) is the U.S. Navy's primary meteorological data and forecasting center. Forecasts are produced from a suite of regional and global numerical models. In light of the proven impact of scatterometer wind vectors on numerical weather prediction (NWP), FNMOC proposed an initiative to receive and assimilate corrected scatterometer wind vector data into one or more of its models. With the availability of wind data from a global model (NOGAPS), FNMOC has also been identified to become the Navy's producer of near real-time corrected wind vector graphical products for Navy operational users. The processing performed at NCEP was chosen as the model for implementation at FNMOC. Once corrected scatterometer wind vectors are being produced on a regular basis, FNMOC will take over the function of producing wind barb graphics from NAVO.

C. RESEARCH OBJECTIVES

As few satellites are designed to operate autonomously, the ground element of any satellite system must assume the responsibility for the command, control, and health of the satellite, including payload control and subsystem and orbit maintenance, and data handling and dissemination of geophysical data products to users. The data processing segment of an operational satellite system begins with the collection of the data by the satellite's instruments, which are then downlinked to the ground station either in real time broadcast or from a storage device onboard the satellite at specified times. At this point, the ground station is tasked with the distribution of the raw data to the users. Often, the ground station also supplies telemetry data with the mission data. The ground station may perform some processing of the data before it is sent to the central-site processing centers, such as FNMOC, from which data are distributed to many users world-wide.

Central-site processing of satellite data into usable geophysical products is influenced by many factors. These include: 1) the physical dynamics of the system of interest, 2) the geophysical relationship between the measurable quantity and the desired parameter, 3) the practical implementation of the geophysical algorithm, 4) the operation of the instrument taking the measurements and the flow of its data to the processing center, and 5) its intended use. An understanding of each is essential to process the data efficiently and effectively. Each of these aspects will be discussed in following chapters as they relate to scatterometer measured sea surface wind data.

The objective of this study is to implement an efficient method of processing ERS scatterometer data at FNMOC using the local global model analysis (NOGAPS) as guidance and to compare results with the ESA-produced wind vectors in the Fast Delivery Product (FDP). In particular, the produced wind vectors will demonstrate the shortcomings of the FDP as illustrated by several test cases, improvements resulting from the inclusion of the NOGAPS analysis, and efficiency in run-time performance. Ultimately, the corrected scatterometer winds will be assimilated into one or more FNMOC models and will be used in the creation of wind vector graphics for dissemination to Navy operational centers.

The premise behind the ability to obtain sea surface wind speed and direction information from backscattered radar signals is discussed in Chapter II. Chapter II also reviews the operation of past and current space-borne scatterometers The chapter concludes with an explanation of the development of the model function currently used to obtain wind vector data from measured radar backscatter. The method of implementation of this model function and its use is discussed in Chapter III. In Chapter IV, the adaptation and revision of this scatterometer processing method at FNMOC is discussed. The results of scatterometer processing at FNMOC as they compare with other sources of sea surface wind data are presented in Chapter V. This thesis concludes with a review of future scatterometer initiatives.

II. BACKGROUND

A. SCATTEROMETER OPERATION

1. History

Early radar data in World War II were contaminated with return pulses from the sea that obscured small vessels and aircraft. In the early 1960's this "sea clutter" and its detection by radar became the focus of investigations by the U.S. Naval Research Laboratory. Wright (1966) established that sea clutter was caused by the effects of a resonant interaction between the radar wave and capillary waves of one-half the wavelength of the radar wavelength. [Ref. 5] Further study revealed the energy in these waves increases with wind velocity. Thus, it was concluded that radar backscatter amplitude increases with the amplitude and density of the capillary waves and, hence, wind velocity.

The determination that backscatter was somehow proportional to the sea surface wind led to a proposal for a satellite-borne instrument and the development of a specialized microwave radar, called a scatterometer. The exploratory development was accomplished under NASA's Advance Applications Flight Experiments (AAFE) program, and the AAFE RADSCAT was subsequently flown on aircraft and the S-193 scatterometer on Skylab in 1973. The results of these experiments led researchers to believe there was adequate potential in the collection of high-quality sea surface wind data from scatterometers to continue development of the instrument. Additional studies concentrated on conversion methods for quantifying the relationship between scatterometer measurements of the area-normalized radar cross-section, or backscatter, and wind speed and direction. [Ref. 5]

On 28 June 1978, NASA launched the first satellite dedicated entirely to ocean surveillance, Seasat. Seasat was launched into a near-circular orbit at an inclination of 108° and an altitude of 790 km. The primary mission of Seasat was to prove the feasibility of global ocean monitoring from space. Seasat was fitted with a number of microwave instruments, including the Seasat-A Satellite Scatterometer (SASS). On October 10, less than four months after launch, the spacecraft suffered a fatal power failure and the mission was terminated. [Ref. 5] Though the planned three-year mission was cut drastically short, the data collected was still studied a decade later and established the instrument as "a breakthrough in maritime meteorology and oceanography [Ref. 6]."

Follow-on missions were already planned at the time of Seasat's failure but were canceled before they were completed [Ref. 6]. No scatterometer flew on a satellite until 1991 with the launch of the European Space Agency's (ESA) European Remote Sensing Satellite, ERS-1. The ERS-2 followed, launched in April 1995. The most recent satellite to launch with a scatterometer on board is the Japanese Advanced Earth Observing Satellite (ADEOS), carrying the NASA-Scatterometer (NSCAT) which operates at a frequency of 13.9 GHz. ADEOS was successfully launched on 16 August 1996.

2. Physical Principles

The physical basis of scatterometer operation is a resonant interaction between radar waves and wind-driven sea surface capillary waves, called Bragg scatter. The resonance condition is a function of the viewing geometry, capillary wavelength and microwave radar wavelength. These parameters are related by

$$\lambda_S \sin\theta = \frac{\sin\phi}{2}\lambda_R, \qquad (1)$$

where l_S is the capillary wavelength, q is the incidence angle, l_R is the microwave radar wavelength and f is the azimuth angle between the wave crests and the radar line of sight. The scatterometer emits energy at wavelengths such that the Bragg scatter conditions are met for wind-driven, 1-4 cm capillary wavelengths. Observation from the aircraft and Skylab experiments demonstrated that greatest wind sensitivity was found with higher radar frequencies, 13.9 GHz being ideal. [Ref. 1]

Radar cross section, normalized by viewing area and represented as σ^o, is anisotropic with respect to the azimuth angle between the wind vector and the radar beam (Figure 1). The maximum return occurs when the radar line of sight is aligned with the wind, with a small difference in σ^o for upwind viewing and downwind viewing; and the minimum occurs when the radar is looking crosswind [Ref. 1]. There is also a polarization dependence, with VV polarized backscatter having higher magnitude than HH polarization. Figure 1 demonstrates the directional response of σ^o at a given incidence angle, showing as much as a 6 dB difference between minimum and maximum values.

Numerous σ^o measurements of one area at different azimuth angles are necessary for the determination of wind speed and direction [Ref. 7], accomplished through a mathematical model which describes the relationship between σ^o, the instrument's incidence angle, and wind speed and direction, and is discussed in detail in subsequent sections of this chapter. This also drives the requirement to radiate signals of narrow beamwidth in azimuth and a beamwidth wide in the vertical to cover the appropriate swath.

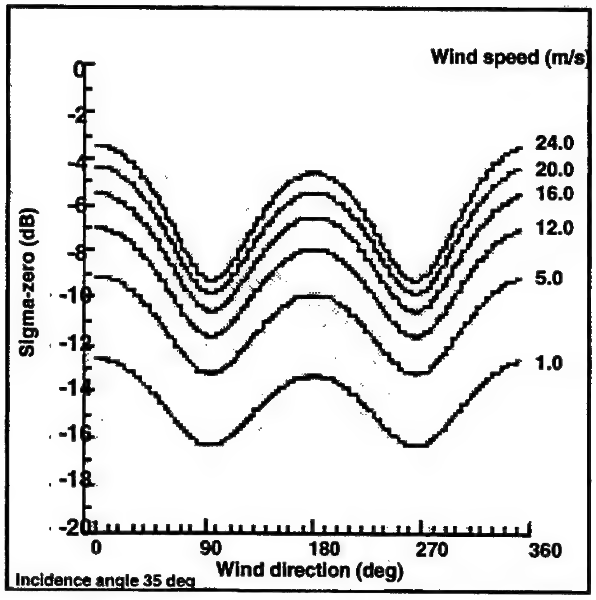


Figure 1. Backscatter (dB) vs. Relative Wind Direction. Data at different wind speeds for 5.3 GHz, vertical polarization, 35deg

These criteria are satisfied with the use of stick antennas that radiate into a fan beam pattern [Ref. 9] with the incidence angle range centered at 40°, operating at a frequency of approximately 14 GHz [Ref. 1]. Within the wide swath illuminated on the surface, the local incidence angle is determined by the range to the measurement point. This range can be calculated either from the Doppler shift of the return signal or from the time delay between transmission and receipt of the pulse [Ref. 9].

B. SPACE-BORNE SYSTEMS

1. Seasat

The first space-borne scatterometer to be flown on a satellite was designed and built for the NASA experimental satellite, Seasat. The Seasat-A Satellite Scatterometer (SASS) consisted of 4 bar-shaped, fan-beam antennas: two forward ($\pm 45^\circ$) and two rear ($\pm 135^\circ$) with respect to the satellite's velocity vector. This allowed the instrument to measure one spot on the surface twice - once with a forward antenna and again 1-3 minutes later with the rear. [Ref. 5] The configuration of the SASS antenna patterns on the sea surface is shown in Figure 2. The instrument operated at 14.6 GHz, corresponding to a Bragg wavelength centered around 1 cm. Each beam illuminated a narrow area on the ground of approximately 0.5° by 25° at incidence angles between 25° and 55°. As the satellite proceeded in its orbit the two antennas on either side covered a swath of 475 km, leaving a gap of 400 km directly under the satellite. A secondary nadir-viewing antenna was also on board and illuminated the area within a $\pm 8^\circ$ zenith angle, covering a swath of 140 km directly beneath the satellite, providing wind velocity magnitude data but not wind direction. [Ref. 1]

All antennas were dual polarized and capable of operating in horizontal-horizontal or vertical-vertical polarization modes. There were eight designated operating modes that included various antenna sequences and polarization. Each sequence included one forward

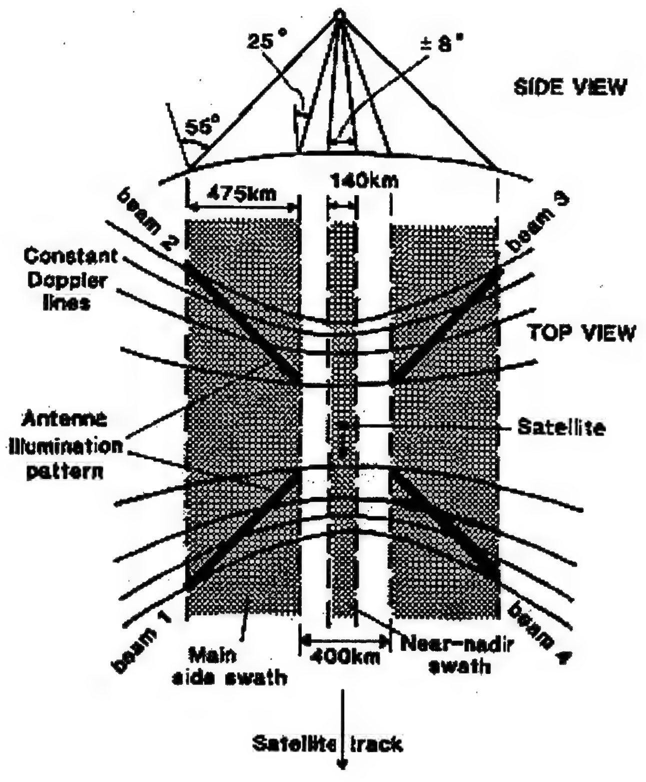


Figure 2. Seasat Scatterometer Swath Geometry. [Ref. 1]

beam and one rear beam on the same side, though only two of the eight operating modes included all four antenna beams. [Ref. 1]

Twelve Doppler filters were used to divide the X-shaped footprint into resolution cells approximately 70 km along the beam and 18 km across [Ref. 5]. This method was used because it was more efficient to transmit the few long pulses needed to measure Doppler shift than the many short pulses required to accurately measure range from time delay [Ref. 9]. Because the view angle varied along the beam, the relative velocity component (in the direction of radar propagation) between the satellite and any point illuminated by the beam varied accordingly. Therefore the reflected signal had a Doppler shift that was a function of the reflection position within the beam. By filtering all but the expected frequency it was possible to resolve spatially the radar return to 50 km. [Ref. 1]

The SASS performed some onboard processing before the data was downlinked to the ground station. A set of algorithms was used to evaluate the gain of the receiving system, which was used to calculate the mean power reflected from the surface during the measurement period. The measurement footprint cell size was determined from the beam geometry and the evaluation of the Doppler pattern relative to the location and satellite direction. From this information, the normalized radar cross-section, σ^{o}, for the particular cell was calculated. Finally, the σ^{o} measurements of the same area by the different antennas were paired together. The data was then downlinked to the ground station for input to the wind measurement algorithm. [Ref. 1]

Processing of the data produced by SASS showed that typically there were four solutions for the wind speed and direction for each pair of σ^{o} measurements. The wind magnitude was approximately the same for each solution, leaving four possible wind directions, the true one and three aliases. In principle, using opposing polarization states should remove the aliases and, accordingly, two of the antenna sequence operating modes did so. However, given the noise in the data, the solution was not so clearly defined and a unique solution has yet to be produced, even using both polarizations. [Ref. 1] From these Seasat analyses, it was concluded that observations from three different azimuth angles could reduce the ambiguity to produce only two possible wind direction estimates 180^{o} apart. [Ref. 9]

2. European Remote Sensing Satellites, ERS-1/2

The European Space Agency (ESA) launched its first remote sensing satellite, ERS-1, in July of 1991 and its follow-on, ERS-2, in April 1995. The satellite carries a complement of instruments designed to use advanced microwave techniques to provide global measurements and images, independent of time of day or weather conditions. The satellite was designed to render fast-delivery data products that "make significant contributions to operational meteorology, sea state forecasting and monitoring sea ice distribution for shipping and offshore activities [Ref. 7]."

To process satellite data, it is important to understand the operation of both the satellite and the instrument. Requirements of the ERS mission include an end-to-end remote sensing system (space and ground segments), worldwide geographical coverage, delivery of standard products within a few hours of acquisition to users, and system calibration and validation campaigns for the system and the resulting data products [Ref. 7].

The satellite bus design was based on the multi-mission French SPOT program. The subsystems of the bus support the payload by providing attitude and orbit control, power, monitoring and control of the payload, telemetry and data handling, structure, and temperature control [Ref. 7]. The payload consists of a core set of active microwave sensors and additional instruments specifically for support to the microwave sensors (Figure 3). The scatterometer is part of the Active Microwave Instrumentation (AMI), which combines the functions of a Synthetic Aperture Radar (SAR) and a wind scatterometer to measure wind fields and wave spectra over ocean and acquires all-weather high resolution images over polar ice coastal zones and land areas. The AMI cannot operate the SAR and the scatterometer simultaneously. As the AMI is in image mode for 12 minutes per orbit, the SAR has approximately a 12% duty cycle. The wind and wave modes of the AMI can operate simultaneously with a 70% duty cycle. [Ref. 3]

The satellite is in a near-polar sun synchronous orbit with an inclination of 98.52°, at an altitude between 777 and 785 km. The mean local time at the ascending node is 2230. ERS-1 has an average orbital period of 100 minutes, resulting in 14.33 orbits per day. [Ref. 7] ERS-1 is a three-axis stabilized earth-pointing satellite, with yaw steering capability to allow for compensation of earth rotation to simplify scatterometer processing. It also allows for geodetic pointing (local normal pointing as opposed to geocentric pointing) of the yaw axis to reduce measurement errors for the radar altimeter. [Ref. 8] Attitude control is provided by a reaction wheel assembly consisting of three reaction wheels and magnetic torque rods to assist in off-loading reaction wheel torques. Additionally, a reaction control unit of hydrazine thrusters is used during attitude acquisition phases and orbital control operations. Power is provided to the power supply subsystem by a sun tracking solar array, the configuration of which can be seen in Figure 3. The satellite has a total mass of 2157.4 kg. [Ref. 7]

Instrument data is handled by the Instrument Data Handling and Transmission (IDHT) which transmits data to ground stations using both X-band and S-band frequencies. The high rate link (Channel 1) for the AMI image mode is done via X-band, as is the low rate link (Channel 2) for both recorded and real-time data produced by the AMI wind and wave modes, the Radar Altimeter and the ATSR/MWS. The S-band is used for telemetry data. [Ref. 7]

The ERS ground segment provides the capabilities for control and operation of the satellite, for reception, archiving and processing of the payload data and for satisfying user product requirements. These services are provided by the network of ground stations consisting of the Earthnet ERS Central Facility (EECF), the Mission Management and Control Center (MMCC), ESA ground stations, national ground stations, Processing and Archiving Facilities (PAFs), and user centers and individuals. The EECF carries out all user interface functions, while the MMCC carries out all satellite operations control and management. The ESA ground stations are located in Sweden, Italy, Canada and the Canary Islands, Spain and provide for data acquisition and the processing and dissemination of fast-delivery products. The national ground stations are located all over the world and provide support for the SAR mission. The PAFs are primarily responsible for archiving data and for generating off-line precision data products. [Ref. 7]

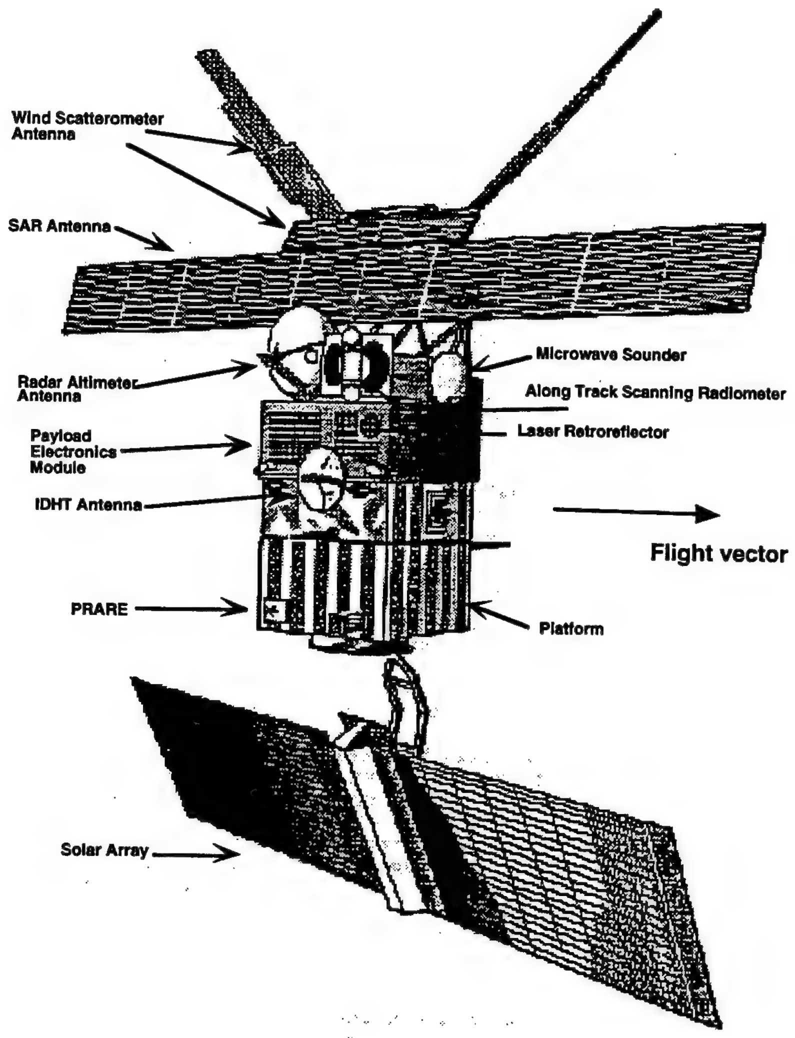


Figure 3. ERS-1 Instrumentation (viewed from Earth, rotated 90°). [Ref. 8]

To allow the oceanographic and meteorological communities to capitalize on the available data from ERS satellites, ESA committed itself to providing Fast Delivery Products (FDPs) to selected sites within three hours of satellite observation. This time constraint, however, limits the accuracy and coverage of the products, hence the PAF production of the off-line precision data products. The FDPs are designed for near real-time use. [Ref. 8] FDPs are available for the SAR image mode, SAR wave mode, Wind Scatterometer, Radar Altimeter and ATSR and are produced at the ground station in Frascatti, Italy.

The Wind Scatterometer on the ERS-1 operates as part of the Active Microwave Instrumentation (AMI) in wind mode. The instrument operates in the C band at a frequency of 5.3 GHz, corresponding to a Bragg wavelength centered around 3 cm, and consists of three antennas that form a swath on the right side of the satellite ground track. The configuration of the scatterometer on the satellite is shown in Figure 3. The antennas are positioned to point at an azimuth of 45°, 90°, and 135° to the satellite velocity vector and are referred to as the fore, mid, and aft beams, respectively. The beams sweep out a swath of 500 km, with the closest point approximately 200 km from the sub-satellite track. The fore and aft antenna beams vary in incidence angle from 25° to 59° as they move across the swath. The mid beam varies from 18° to 47° in incidence. A point within the swath is eventually illuminated three times, once by each antenna. Figure 4 displays the scatterometer antenna and beam geometry, and instrument specifications are contained in Table 1.

The ERS instrument differs from the SASS in that the antennas are only on one side of the satellite; it measures in C band rather than the Ku-band used by the SASS; and, more importantly, there are three antennas at unique azimuth angles to illuminate each node three times for improved wind direction retrieval. Additionally, the signal processing of the ERS-1 scatterometer is by the more conventional short-pulse range gating, rather than the long-pulse Doppler-gating used by the SASS. [Ref. 7] Pulse range gating uses short pulses to estimate the distance between the instrument and the measurement point using the time delay between transmission and receipt. The short pulses experience little Doppler shift as opposed to the long pulses used in range resolution by the SASS.

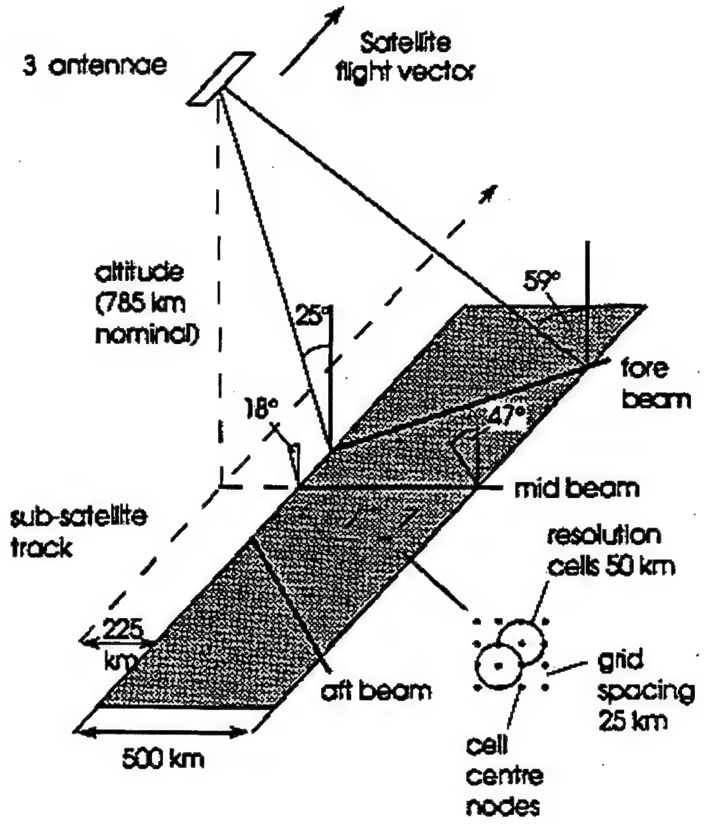


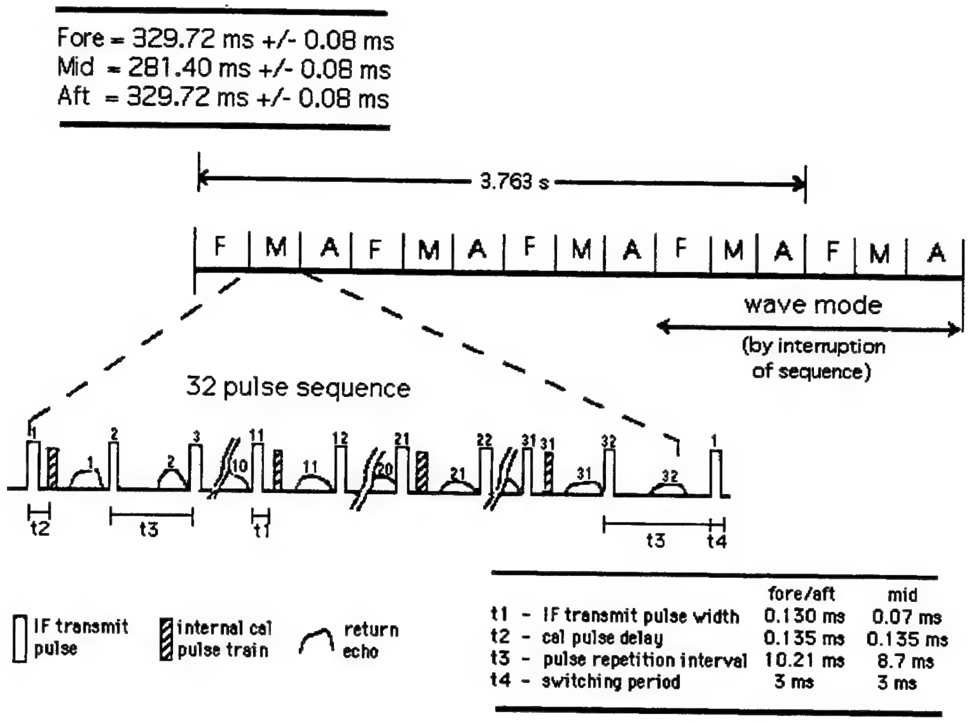
Figure 4. ERS Scatterometer Swath Geometry. [Ref. 7]

The scatterometer illuminates the measurement nodes by RF pulses (peak power) from each antenna. Each measurement node is separated by 25 km and is centered in a 50 km x 50 km resolution cell, resulting in 19 measurement points across the 500 km swath, as seen in Figure 4. The location of each node is determined by the range gating method mentioned above. The effect of the Earth's rotation is compensated for by the satellite's control of the yaw axis. The pulse is produced at the Intermediate Frequency (IF) and is then amplified, converted to an RF signal and amplified again. The signal is then sent to the appropriate antenna by the circulator assembly, which is controlled by the converted, amplified, and routed to the instrument electronics. The 3.763 seconds it takes to complete the measurement sequence corresponds to 25 km along the sub-satellite track when the satellite is at an altitude of 785 km and is repeated continuously. As the satellite altitude varies over a range from 769 km to 825 km, however, this fixed measurement sequence timing does not correspond exactly to 25 km. [Ref. 7]

The 3.763 second measurement sequence allows for four series of measurements at a constant rate by each the fore, mid, and aft antennas and is shown in Figure 5 with the fore, mid, and aft beams noted as F, M, and A respectively. This corresponds to 32 measurement pulses on each beam. Internal calibration and noise measurements are also

Table 1. Wind Scatterometer Technical Specifications, After [Ref. 7]

Frequency:	5.3 GHz ± 52 kHz		
Polarization:	Linear Vertical		
Peak power (RF):	4.8 kW		
	FORE	MID	AFT
Antenna aspect angle:	+45°±0.5°	0°±0.5°	-45°±0.5°
Antenna length:	3.6 m	2.5 m	3.6 m
Pulse width:	130 μs	70 μs	130 μs
No. of pulses per 50 km:	256	256	256
Radiiometric resolution (Kpe):	9.7%	8.5%	9.7%
Detection bandwidth:	25 kHz	25 kHz	25 kHz
Sampling scheme:	-------- Complex I/Q 8 bits each --------		
Return echo window duration:	3.93 μs	2.46 μs	3.93 μs
Incident angle range:	25°-59°	18°-47°	25°-59°
Singal-to-noise contribution:	282.0 dB Hz		
Uncompensated gain stability:	4.1 dB / 4.8 dB		
Compensated gain stability:	0.3 dB / 0.48 dB		
Calibration pulse delay:	135 μs		
Spatial resolution:	≥ 45 km (along and across track)		
Spectral resolution:	≥ 1/55 km (along and across track)		
Radiometric resolution			
4 m/s:	≤ 8.5% (mid beam), ≤9.7% (fore/aft beam)		
24 m/s:	≤ 6.5% (mid beam), ≤7.0% (fore/aft beam)		
Cross polarization:	≥ 15 dB		
Swath width:	≥ 500 km		
Localization accuracy:	±5 km (along and across track)		
Wind direction range/accuracy:	0-360° / ±20°		
Wind speed range/accuracy:	4 m/s - 24 m/s / 2 m/s or 10 %		


Fore = 329.72 ms +/- 0.08 ms
Mid = 281.40 ms +/- 0.08 ms
Aft = 329.72 ms +/- 0.08 ms
3.763 s
F M A F M A F M A F M A F M A
wave mode
(by interruption of sequence)
32 pulse sequence
t2
t3
t1
t3
t4
IF transmit pulse
internal cal pulse train
return echo

	fore/aft	mid
t1 - IF transmit pulse width	0.130 ms	0.07 ms
t2 - cal pulse delay	0.135 ms	0.135 ms
t3 - pulse repetition interval	10.21 ms	8.7 ms
t4 - switching period	3 ms	3 ms

Figure 5. Wind Scatterometer Measurement Sequence.

regularly performed in the window between the transmit of the pulse and its return. As the Doppler variation is considerable for the fore and aft beams over the swath (20 kHz near swath to 140 kHz far swath), a programmable Doppler compensation law is applied to the returned signal to these antennas prior to filtering and complex sampling. [Ref. 7]

When the measurements are taken, they are stored on-board for transmission to one of the ground stations each orbit. Once at the ground station, the raw radar return is used to calculate a σ^{o} value for each antenna for each 50 km x 50 km resolution cell. This is accomplished in terms of other known (or measurable) parameters using

$$\sigma^{\circ} = \frac{P_r}{P_t} \frac{64\pi^3 R_s^4}{\lambda^2 L_s G_0^2 \left(\frac{G}{G_0}\right)^2 A}, \tag{2}$$

where P_t is the transmitted and P_r the received power, R_s the slant range to the target of area A. Radar wavelength is represented by λ, and L_s includes atmospheric attenuation and other losses. Peak antenna gain is G_0, and G/G_0 is the relative gain in the target direction. [Ref. 8] The σ^o triplets are then processed in the Fast Delivery Product (FDP) format and distributed within three hours from instrument observation. [Ref. 7] Unlike the SASS which often produced four possible solutions, the ERS-1 scatterometer, with its three antennas, typically produces two solutions with approximately the same wind speed, and directions that generally are 180° apart.

C. MODEL TRANSFER FUNCTION

1. Development

To take advantage of the global potential of the space-based scatterometer, there must exist a method to accurately derive both wind speed and direction from the instrument-measured backscatter. Wind retrieval from σ^o measurements is accomplished by two major steps, beginning with a relationship between backscatter and wind speed and direction, called a model transfer function. A scatterometer model function is based on the fact that σ^o displays a power law relationship to wind speed of the form

$$\sigma^\circ = C_1 U_{10}{}^{C_2}, \tag{3}$$

where C_1 and C_2 are both a function of incidence angle, azimuth angle relative to wind direction, and polarization and U_{10} is the neutral wind speed at a height of 10m (Figure 6) [Ref. 5]. The scatterometer model function is of the form $\sigma^\circ = f(U_{10}, \phi)$, where U_{10} is as defined above and f is the wind direction relative to the antenna look angle. "Inversion" of equation (3) is then used to determine wind speed and direction given σ^o measurements. The second step is called ambiguity removal and is necessary because the model function produces multiple solutions. The ambiguity removal is performed to select the most probable solution. [Ref. 10]

Wind retrieval has generally been accomplished by empirical relationships derived from actual measurements, as was the case for the SASS-1 transfer function developed for

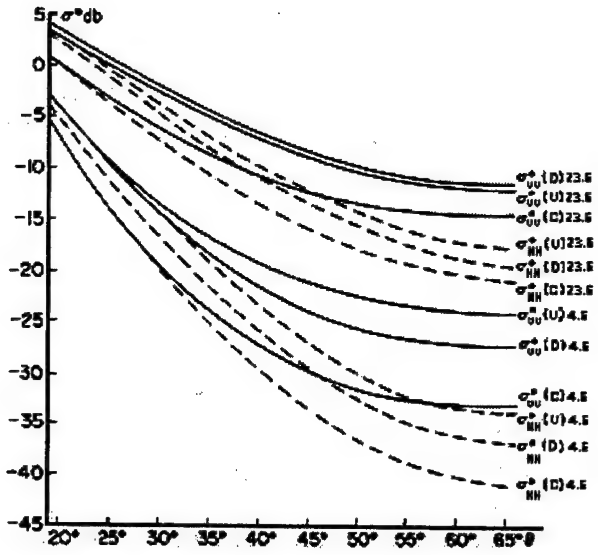


Figure 6. Backscatter vs. Incidence Angle (for SASS-1 model function) for winds of 23.6 and 4.6 m/s. [Ref. 1]

Seasat. Because backscatter is a function of radar frequency, the SASS-1, developed for the Ku-band (14.6 GHz) SASS, is not valid for use with the ERS-1 C band (5.3 GHz) scatterometer. In preparation for the launch of ERS-1, ESA began several campaigns in the mid-1980's to collect collocated airborne scatterometer data and wind observations, all with a reference height of 10m. From this effort an empirical model at C band, known as CMOD1, was derived. Data from subsequent campaigns resulted in an improvement to CMOD1, yielding the pre-launch transfer function CMOD2 which was used for wind retrieval until June 1992. [Ref. 11]

CMOD2 is of the form

$$\sigma^\circ = 10^\alpha U_{10}^\gamma (1 + b_1 \cos\phi + b_2 \cos 2\phi), \qquad (4)$$

where the coefficients b_n, a, and g are dependent on incident angle. Relative wind direction is defined such that f=0° occurs when looking upwind, f=180° when looking downwind, and f=90° or 270° when looking crosswind. [Ref. 11] An example of this relationship is shown in Figure 1, illustrating σ° as a function of f for various constant wind speeds.

With the launch of ERS-1, however, there was concern that CMOD2, developed from data gathered by aircraft scatterometers, would not be valid for the ERS-1 scatterometer, primarily due to the large difference in footprint size for the two types of instruments [Ref. 10]. Of principal importance was to meet the product specification for the retrieved wind speeds to 2 m/s rms and wind direction to 20° over the range 4-24 m/s. Consequently, the RENE-91 campaign was launched off the coast of Norway in the last three months of 1991, to validate the ability of CMOD2 to predict the σ^o values measured by the ERS-1 scatterometer and to retrieve the appropriate winds. Aircraft, buoy, ship, and model data were compared with the ERS-1 scatterometer winds produced by CMOD2. Comparison of the analyzed RENE-91 data with ERS-1 winds from CMOD2 showed rms differences of 2.7 m/s in speed, 103° in direction prior to ambiguity removal and 19° after ambiguity removal, 4.1 m/s for the vector, and 62% of directions within $\pm 90^\circ$ of the analysis for ambiguity removal [Ref. 11]. These results confirmed the need for further tuning if the product specification was to be met.

The next model adopted by ESA (CMOD3), implemented in June 1992, was developed by the European Center for Medium-Range Weather Forecasts (ECMWF). Development of CMOD3 began with an investigation into the characteristics of the σ^o measurements. Since any phenomenon that affects the ocean surface could alter the capillary waves, σ^o could be a function of parameters other than the local wind velocity, such as temperature, surfactants, or wave steepness. It was important to determine what parameters have substantial effect on σ^o measurements before beginning work on a new two-parameter transfer function. One approach to this task was to plot σ^o from the model function in 3D measurement space. If a transfer function is dependent on only two parameters, the resulting data should lie on a surface in 3D space. The σ^os spanned by the fore, mid, and aft beams were plotted, producing the surface shown in Figure 7. It was consequently determined that the effect of parameters other than the local wind vector on σ^{o} measurements are secondary, if present at all, and accurate wind retrieval is possible based solely on a σ^{o}-to-wind relationship. [Ref. 10]

Stoffelen and Anderson [Ref. 10] validated the σ^{o} triplet distribution in Figure 7 as a cone-shaped surface composed of two closely overlapping sheaths. CMOD2 did not fit

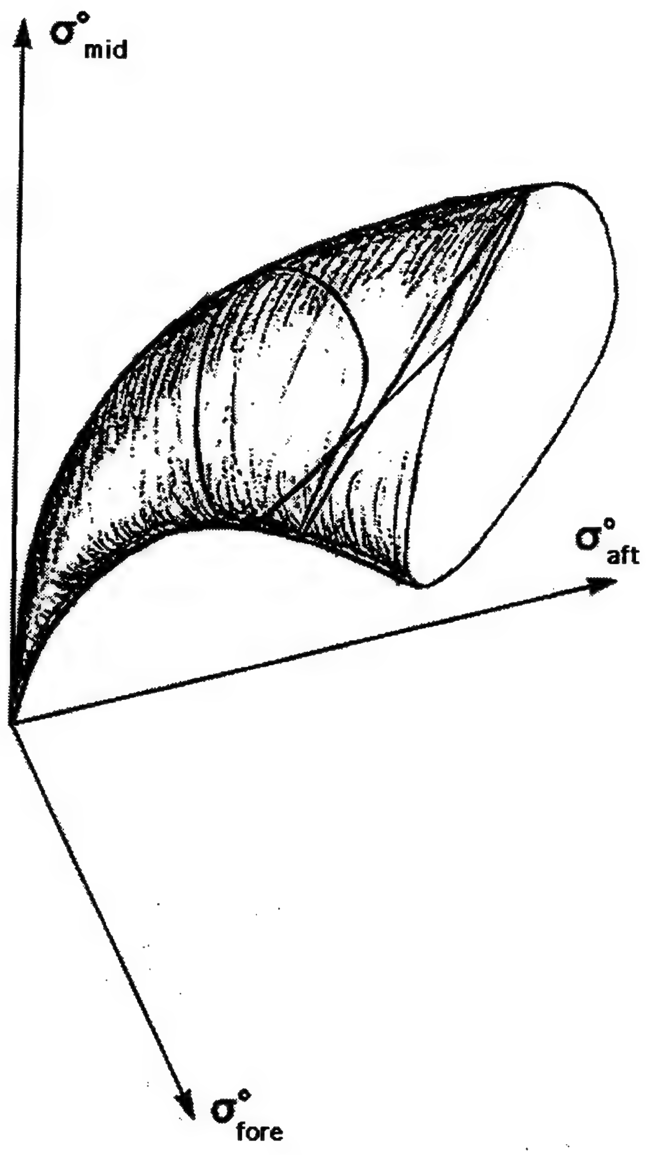


Figure 7. Representation of σ° Triplet Measurement Surface for a Given Node. [Ref. 10]

this distribution in the measurement space to adequate accuracy. The CMOD3 model function was subsequently developed to fit the σ^o space within measurement error specifications. Following an engineering calibration, however, deficiencies in wind retrieval were noted at small incidence angles and low wind speeds [Ref. 11]. Further tuning by ECMWF produced the CMOD4 model, implemented in February 1993 and still in use by ESA and other centers for scatterometer wind retrieval. CMOD4 is of the form

$$\sigma_{lin}^{0} = b_0(1 + b_1 \cos\phi + b_3 \tanh b_2 \cos 2\phi)^{1.6} \tag{5}$$

and is addressed in detail in Appendix A.

2. Deficiencies

In the formulation of CMOD4, a study of the σ^o measurement space revealed scatter of data from the expected surface displayed a wind speed dependency for low wind speeds. It has also been shown that the variance of the scatter is larger for small incidence angles (high σ^o values) than that for large incidence angles (low σ^o). This is possibly due to the non-linear dependence of σ^o on incidence angle (Figure 6), and the fact that wind and waves vary on a scale smaller than that of the scatterometer effective footprint. This non-linearity results in greater sensitivity for the scatterometer at the inner part (lowest incidence angle) of the footprint. The effective footprint, therefore, differs for each beam, particularly for the inner measurement nodes (low incidence angles). Consequently, geophysical variability on scales smaller than that effectively sampled will increase scatter and, therefore, error in the returned wind vector. [Ref. 10]

In general, the CMOD4 transfer function fits the reference σ^o space quite well. The scatter distance of measured data from the expected surface previously described is generally on the order of the instrument noise (~5%). Comparisons of CMOD4 scatterometer winds with ECMWF first guess winds by Stoffelen and Anderson [Ref. 10] showed that the greatest scatter of the scatterometer winds occurred within 100 to 150 km of extreme weather conditions, such as intense fronts and low pressure centers. This most likely results from effects of geophysical parameters other than wind, such as rain or sea state, or wind variability on a scale smaller than the effective footprint. This apparent dependence on geophysical parameters other than wind is not expected to generally affect the global quality of the wind [Ref. 10]; yet, it is these areas of high wind variability that are of greatest interest to the users of this data.

Finally, questions have been raised as to the ability of CMOD4 to return high winds. In the derivation of CMOD4, a uniform distribution of wind speed was used; however, speeds greater than 15 m/s were undersampled. Stoffelen and Anderson [Ref. 10] performed comparisons with model winds for areas with high wind speeds. These comparisons typically showed higher speeds produced by the model than by CMOD4. Though wind speeds of up to 22 m/s were retrieved, on occasion, from CMOD4, the range of validity is considered to be 4-18 m/s for the transfer function. Furthermore, comparisons performed by Pierson and Sylvester [Ref. 12] of CMOD4 with high measured winds show that the required backscatter values for the wind speeds are too high and, therefore, would not be correctly recovered by CMOD4. Work continues on the tuning of the transfer function.

III. ERS SCATTEROMETER DATA PROCESSING METHOD

A. INTRODUCTION

The processing scheme developed by the National Center for Environmental Prediction (NCEP) is the basis for the method chosen for implementation at the Fleet Numerical Meteorology and Oceanography Center. NCEP retrieves the data from National Environmental Satellite, Data, and Information Service (NESDIS) and processes the σ^o triplets locally. The wind scatterometer Fast Delivery Product includes the ESA produced wind speed and direction and the calculated σ^o triplets for each node. The information contained in the FDP is given in Table B.1, Appendix B. For dissemination, the ESA groups the data in "products" of 361 measurement nodes, comprised of 19 measurements across each of 19 swaths (approximately 500 km x 500 km), as shown in Figure B.1 of Appendix B, and are assigned the same measurement time. The product is sent to NESDIS via the UK Met Office.

Processing the raw scatterometer data requires two primary steps: quality control and wind retrieval. Valid measurements are considered those taken over open ocean and exceeding levels of quality in measurement integrity. The process of retrieving wind vector information from σ^o measurements consists of inversion and ambiguity removal. Inversion is the use of a model transfer function to determine wind speed and direction from σ^o measurements. Ambiguity removal is needed to choose the most likely of multiple directional solutions and is usually accomplished with the aid of an auxiliary data source for comparison.

This chapter discusses the method of quality control designed and used by NCEP [Ref. 13] and the implementation of the wind retrieval algorithm designed by the United Kingdom Meteorological Office (UK Met Office) and adapted by NCEP [Ref. 14].

B. NCEP PROCESSING OVERVIEW

NCEP receives ERS scatterometer data from NESDIS in six hour increments encoded in Binary Universal Format (BUFR). The files are decoded and processed for assimilation in the NCEP numerical weather prediction (NWP) models. The processing scheme is comprised of three primary FORTRAN 77 routines that decode and quality check the data and then process the data to produce wind vectors, Figure 8. The wind retrieval algorithm, adapted from one developed at the UK Met Office by Offiler, consists of three parts: inversion of the σ^o to wind speed and direction and ranking of the ambiguous solutions, re-ranking of the wind vectors by comparison to a background field, and smoothing of the most likely wind vectors by a five by five median filter which performs a buddy check, called Sequential Local Iterative Consistency Estimator (SLICE).

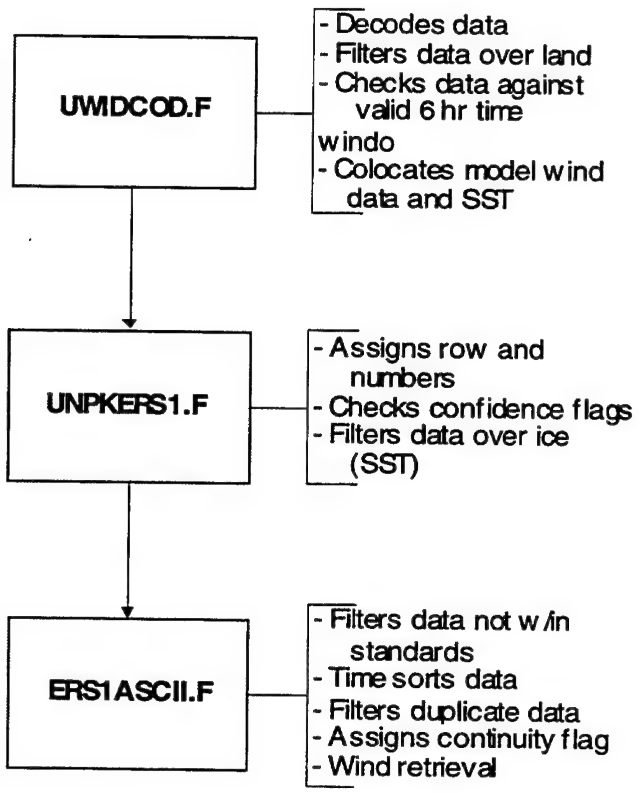

UWIDCOD.F
- Decodes data
- Filters data over land
- Checks data against valid 6 hr time windo
- Colocates model wind data and SST
UNPKERS1.F
- Assigns row and numbers
- Checks confidence flags
- Filters data over ice (SST)
ERS1ASCII.F
- Filters data not w/in standards
- Time sorts data
- Filters duplicate data
- Assigns continuity flag
- Wind retrieval

Figure 8. NCEP Scatterometer Processing Flow Diagram (as of December 1995).

NCEP processes the data every six hours, in accordance with the NWP model runs, 0000, 0600, 1200, and 1800 Greenwich Mean Time (GMT) daily. Model data from either the current analysis or the appropriate six-hour forecast are needed for both quality control and as the background wind field used for the ambiguity removal process. Timeliness is a concern when comparing the measured data to model data. Therefore, execution of the scatterometer processing for each synoptic time begins at 0200, 1100, 1400, and 2200 GMT, respectively. More detailed information on the processing schedule at NCEP is available from Peters, et al [Ref. 13].

C. QUALITY CONTROL

Quality control is performed to filter data not appropriate for wind retrieval. Such data is either not over open ocean (i.e., over land or ice), does not have a complete triplet of σ^o, or there is a lack of confidence in the integrity of the measurement, as represented by ESA's 12 bit confidence flag that accompanies the data (Table 2). Each bit represents information about the ESA calculated FDP wind vector and the measurement with which it is associated, as determined by the receiving ground station. Of primary importance are the first three bits of the flag, representing the presence of a σ^o measurement for the fore, mid, and aft beam, respectively.

The first FORTRAN routine, UWIDCOD, handles the data one record (measurement node) at a time and begins by decoding the record with a local BUFR decoder. The first quality screen is for land contamination, as compared to a land mask file used by NCEP. Those nodes not over land are then checked against the valid time window ($\pm$ 3 hours) for the most recent model analysis or forecast. Data which are too old are not processed further. NCEP model-produced sea surface temperature (SST) and wind vector data are then collocated with the scatterometer data records. The record and collocated data are then written to an intermediate file.

Table 2. ESA Fast Delivery Product Confidence Flag.

Bit	Meaning when set
1	No forebeam calculation
2	No midbeam calculation
3	No aftbeam calculation
4	Forebeam arcing detected
5	Midbeam arcing detected
6	Aftbeam arcing detected
7	Any Beam Kp above or equal to threshold
8	Land (any land in cell footprint)
9	Autonomous ambiguity removal not used
10	Meteorological background not used
11	Minimum residual exceeded threshold
12	Frame checksum error detected
All 13	Missing value

The next FORTRAN routine, UNPKERS1, assigns a row and cell number to each node, ranging from 1 to 19 and representing the location of the node across the swath, where 1 is closest to nadir and 19 is the furthest crosstrack point along the swath. The row number also ranges from 1 to 19 and represents each of the swaths comprising a data product previously described. Row numbers begin with 1 as a new product is processed. The cell numbers are assigned according to incidence angle. Following this, the confidence flag table is checked. If any of the bits relating to the σ^omeasurements are set, the record is not processed further. The collocated SST data is then used to check for ice contamination. If the temperature is less than 0° C, it is assumed that ice is present, the measurement is contaminated, and the node is filtered from further processing. The record is then written out to a second intermediate file.

When all valid records are written to the file, the final routine, ERS1ASCII, begins by reading in the records and assigning a block number to the data. A block is considered a 19 row by 19 cell data product discriminated by time, as all data in a block have the same measurement time. Each record is then checked against more restrictive standards than those in the confidence flag table. A record is filtered if the noise in the signal (Kp) from a single antenna exceeds 10% or if the total missing pulse count is greater than 15 (out of 32 for each antenna). Next each block is time sorted and duplicate blocks are filtered. Chronological and overlap problems can occur because data come from different ground stations around the world and may contain overlapping times. The last step prior to wind retrieval is assignment of a continuity flag to the blocks of data. Blocks are considered continuous if the difference in measurement times is less than 1.5 minutes. This is done in preparation for SLICE, a 5 x 5 node array filter that performs a buddy check of the data for ambiguity removal. At this point, the "good" data have been re-dimensionalized by the spatial row and cell numbers and are ready for the wind retrieval algorithm.

D. WIND RETRIEVAL

1. Inversion

Theoretically, it should be possible to retrieve wind speed and direction from σ^o measurements and the transfer function using appropriate simultaneous equations. It was discovered with SASS data, however, that measurement error and the non-linear nature of the function make this straight-forward approach unfeasible. Instead a least-squares fitting approach is used to minimize a difference function comparing measured σ^os and those from the transfer function, using an estimate of wind speed and direction. [Ref. 8] For the SASS, this process generally produces four solutions. With ERS-1, however, measurements from the three antennas most often result in only two possible solutions.

For the ERS scatterometer, wind retrieval using CMOD4 is accomplished numerically by minimizing a difference function of the form:

$$R = \sum_{i=1}^{3} \left(\frac{\sigma_i^0 - \sigma^0(v, \phi_i, \theta_i)}{\sigma_i^0 \cdot Kp_i} \right)^2, \tag{6}$$

where σ_i^0 is the measured value; $\sigma^0(v, \phi_i, \theta_i)$ is the value determined by the transfer function (CMOD4) from an estimated wind speed and direction; Kp is the noise factor in each beam and is a function of the signal-to-noise ratio and system bandwidth; and R is the sum of the squares of the residual values. The wind vector estimate used to find $\sigma^0(v, \phi_i, \theta_i)$ is adjusted to minimize R. The function can converge on as many as four apparent solutions, all with similar speed but different directions. The ERS-1 scatterometer, with three antennas, generally produces only two distinct directions, typically 180° apart. [Ref. 8] This is directly related to the fact that the "upwind" and "downwind" sheaths of the cone in the σ° measurement space are closely overlapping; i.e., the difference between the upwind/downwind σ° maxima is very small (Figure 7) [Ref. 10]. Other solutions, if determined, are usually approximate duplicates of the first two. The solutions are ranked in order of increasing residual.

In 1989, D. Offiler of the UK Met Office developed computer code for operational scatterometer processing at the ECMWF (since updated for CMOD4) [Ref. 14]. This code performs both inversion and ambiguity removal (Figure 9). For the purpose of inversion, look up tables (LUT's) were generated "off-line" from the CMOD4 transfer function, a quadratic function, and derivatives of that function. "CMOD_QSLUT" contains pre-computed quadratic coefficients that give log(speed) as a function of backscatter (in dB), incidence angle, and wind direction relative to the beam azimuth. The second LUT used is specified by the logical name "CMOD_DBLUT," and interpolates σ° (in dB) from a given wind speed (1-30 m/s, 1m/s increments), wind direction (0° -180°, 5° increments), and incidence angle (15°-60°, 1° increments).

The wind retrieval code flow diagram shown in Figure 9. Provided here is a description of the method employed by the code and an example of its use is given in Appendix C. The goal in the inversion step of the wind retrieval process is to minimize equation (6) and is performed in the FORTRAN subroutine WINRET. Minimization requires a value for σ° computed from CMOD4 using an estimated wind speed and direction. The first task performed is to step over 360° in wind direction in 15°

increments (with overlap). At each step, the function QSPEED calculates the speed from coefficients obtained from CMOD_QSLUT using the chosen wind direction, σ^o triplet, and incidence angle. Also at each step, the function RESID is used to calculate the sum of the square of the residual difference between the measured backscatter for each beam and the theoretical backscatter for the given cell geometry. The theoretical backscatter value is determined in the function TSIGDB. In this function, CMOD_DBLUT is used to interpolate the appropriate backscatter value from the given wind direction, estimated speed, and incidence angle. This produces the theoretical $\sigma^0(v,\phi_i,\theta_i)$ value required for equation (6). The sum of the square, or Root Mean Square (RMS) value, of the residual from each beam is computed in this manner for each wind direction over 360°.

The subroutine FIT3 then searches for the local minima over all wind directions. A quadratic is fitted to each minimum and its two neighboring residuals to estimate the actual

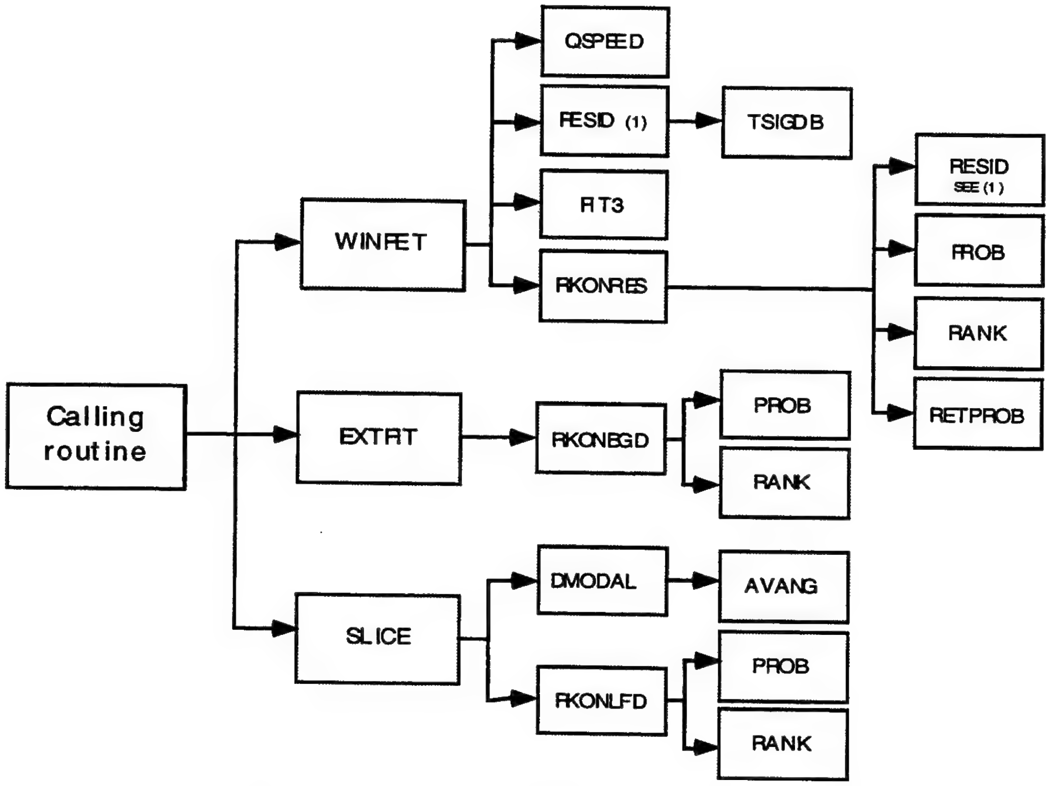


Figure 9. Wind Retrieval Flow Diagram.

position (wind direction) of each minimum. The speed is then estimated for the new direction using QSPEED. This process is repeated until all possible directions are searched or four minima are found. No more than four solutions are calculated, and there must be at least two solutions found for the node to be processed further. This is to allow for the ambiguity removal process (described in the next section) to be used to make a more educated guess of the correct wind vector.

Following this procedure the subroutine RKONRES assigns the solutions a probability value. This probability is calculated by the function PROB from the residual value, recalculated for each solution using RESID, and the estimated standard deviation in the σ^o measurements (assumed Gaussian, with zero mean difference). The solutions are then ranked in order of relative probabilities by the subroutine RANK. Due to the deficiencies of CMOD4 (non-linear dependence of σ^o on incidence angle, increased sensitivity in the inner part of the footprint) the probability of the first ranked solution does not necessarily indicate the correctness of the solution. Therefore, the subroutine RETPROB calculates a pre-validated substitute probability value in accordance with the specific behavior of CMOD4, using the number of solutions retrieved, the mid beam incidence angle, and the first ranked wind direction for the calculation.

2. Ambiguity Removal

Once the candidate solutions are determined, the most likely of those solutions to be closest to the true wind vector must be chosen for the data to be of practical use. The solutions have been assigned probability values based on the CMOD4 transfer function and the measurements themselves. There is an inherent ambiguity in the direction of the retrieved winds, however, and it has proven necessary to perform additional processing to select the best solution. This might be accomplished by selecting areas with directional consistency and adjusting the ranks of neighboring cells which do not agree with this trend.

The danger in this method arises when the areas of consistency are ranked incorrectly and the wrong direction is propagated throughout the wind field. It becomes important, then, to have an idea of the "true" direction prior to using such a method. This is most often accomplished with ancillary data such as ship or buoy measurements or model first-guess (analysis) or forecast fields, which can improve the skill of the wind retrieval by more than 25%. This skill, however, is completely dependent on model accuracy.

The first step in the selection of the "correct" wind vector utilizes the NCEP model wind fields that have been interpolated to the scatterometer measurement nodes and is performed in the subrutine EXTFIT. This step begins by calling the subroutine RKONBGD which computes the standard deviation in both the scatterometer and model wind vectors. Next each scatterometer solution is compared to the model wind vector for the node. The probability of each solution being correct as compared to the model wind is calculated using the standard deviations of each in the function PROB. These probability values are used to modify the probabilities calculated in the inversion stage. The solutions are then re-ranked by the function RANK according to the new probability values.

The last step in the wind retrieval process is SLICE, a filter that performs a "buddy check" on each node by comparing it to the surrounding nodes in a 5 x 5 node array format. The filter sweeps along and across the swath in alternate directions, beginning with the first row at the outer edge of the swath. It continues until fewer than 10 solutions have been re-ranked or six interations have been completed. SLICE handles data in continuous sets, determined by the continuity flag set at the end of the quality control stage of processing discussed earlier.

The process begins by gathering neighboring rank 1 wind vectors within a 5 x 5 block, excluding cells of low rank 1 relative probability (< 0.5) and the center cell (the one being compared). The function DMODAL is then used to estimate the general wind direction from the cells in the 5 x 5 block. If there are fewer than four cells in the array, no local direction is derivable, the value is set to a bogus number, and the cell is temporarily skipped. For the valid 5 x 5 node arrays, all possible wind direction angles (0-360°) are divided into 45° sectors and the number of rank 1 solutions with wind directions in a sector are added and weighted by their solution probability values. For clarification refer to the example in Appendix C. The sector with the greatest weighted sum is added to its two neighboring sectors. If the total number of direction angles in the three sectors is less than four, there are not enough to assume a local consistency, and the estimated local direction value is set to a bogus number. If there are at least four direction angles, the average of the angles in the three sectors, calculated by the function AVANG, is assumed to be the best estimate of the local direction. An ambiguity removal confidence update factor for each node is then calculated as a function of the degree of consistency.

Finally, the subroutine RKONLFD is called to modify the probability of each solution to fit the estimated local direction and to re-rank the solutions accordingly. If the local direction is equal to the bogus number (assigned if there are not enough cells in the array or if there is no consistency) the cell is abandoned for the time being. The re-ranking is accomplished with the ranking routines used previously, PROB and RANK, comparing all solutions with the estimated local direction. The new order of the solutions for each cell is saved until the filter passes the cell again. When fewer than 10 ranks have been changed, the filter is finished with the set of data and returns to begin work on the next set. Following SLICE the data is written to an output file for future use.

IV. FNMOC IMPLEMENTATION

A. INTRODUCTION

Before implementing the NCEP scatterometer processing scheme into the operational framework at FNMOC, it was necessary to determine whether the wind vectors produced by this alternative method are enough of an improvement over the FDP to justify the time used in their production.

The first step taken to answer this question was to use the code developed by NCEP as a basis for executing the same process at FNMOC. After receiving the code from NCEP in December 1995, the code's efficiency was increased by taking advantage of the array processing ability of the Cray supercomputer on which it runs. In addition to processing the data as quickly as possible, logical coding and proper documentation are necessary to assist maintenance while in an operational environment. Therefore, the quality control portion of the code was rewritten. The wind retrieval portion produced by the UK Met Office has essentially been unchanged. This chapter discusses the development of the process in use at FNMOC.

B. FNMOC PROCESSING OVERVIEW

FNMOC's resources and system configuration played an important role in the design of the processing code and the deviation from the code developed by NCEP. The FNMOC code differs from that written by NCEP primarily in the handling of data (array format versus record by record) and the program flow. The FNMOC code follows a more efficient flow with regard to computer system interface. The code also employs the Ocean Model Support Program (OMSP), a locally produced compilation of FORTRAN routines that perform such functions as converting coordinate systems, interpolating data to a specific set of points, and others that are particularly useful [Ref. 15]. As FNMOC is the center of expertise on the Defense Meteorological Satellite Program (DMSP) Special Sensor Microwave/Imager (SSM/I) passive radiometer, a locally produced SSM/I ice analysis is used in the quality control portion of the code as opposed to the sea surface temperature data used by NCEP. Research conducted at the European Center for Medium-Range Weather Forecasts (ECMWF) found that SSM/I derived sea ice data were more "stable and realistic" than ice masks derived from SST data [Ref. 16]. Wind vector data from FNMOC's global model, NOGAPS, are used as the background field in the wind retrieval phase. NOGAPS runs every six hours at 0000, 0600, 1200, and 1800 Greenwich Mean Time (GMT) daily. Data received by FNMOC and data produced by the models are stored for a limited period of time in the Integrated Stored Information System (ISIS) database, accessible on most local platforms.

It should be noted, during the course of this work the source of scatterometer data shifted from ERS-1 to ERS-2 on 3 June 1996. ERS-2 has the same bus design, mission objectives and instrumentation as ERS-1 [Ref. 3]. Following the transfer of responsibility, the scatterometer on ERS-2 became unavailable on 15 July 1996 due to a power anomaly. The scatterometer electronics and the Calibration Subsystem DC/DC converter were switched to redundant systems and the instrument resumed operation on 26 July. No noticeable effect on the data has been detected, however, tests continue to confirm this conclusion. No further anomalies have been detected.

C. PROCESSING CODE

ERS scatterometer data are retrieved from NESDIS in Binary Universal Format (BUFR) format as received from ESA, differing from NCEP which requires specially formatted six-hour files. The data are in the standard "product" format, consisting of 500 km x 500 km blocks divided into 19 rows and 19 cells. Data are then decoded using a universal BUFR decoder obtained from NESDIS and moved to a directory on the Cray

J90 mini-supercomputer for processing. The code at FNMOC was written using a modular approach, resulting in one main FORTRAN routine (ERS1SCAT) which calls eight primary subroutines, as shown in Figure 10.

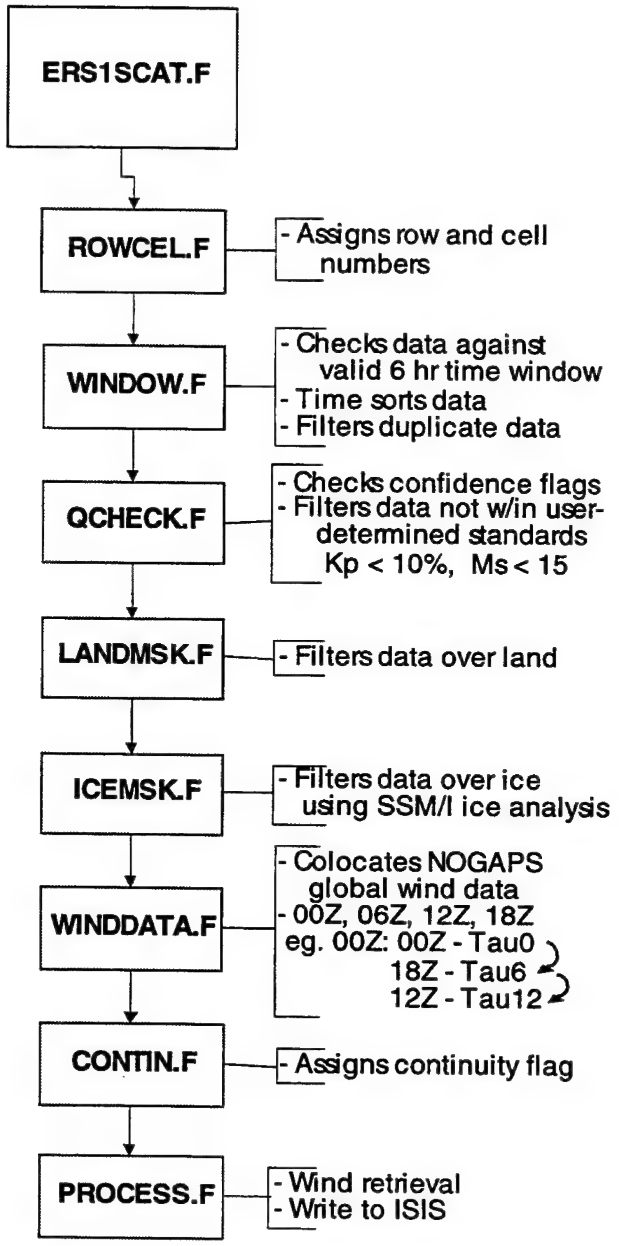

ERS1SCAT.F
ROWCEL.F
- Assigns row and cell numbers
WINDOW.F
- Checks data against valid 6 hr time window
- Time sorts data
- Filters duplicate data
QCHECK.F
- Checks confidence flags
- Filters data not w/in user-determined standards Kp < 10%, Ms < 15
LANDMSK.F
- Filters data over land
ICEMSK.F
- Filters data over ice using SSM/I ice analysis
WINDDATA.F
- Colocates NOGAPS global wind data
- 00Z, 06Z, 12Z, 18Z
eg. 00Z: 00Z - Tau0
18Z - Tau6
12Z - Tau12
CONTIN.F
- Assigns continuity flag
PROCESS.F
- Wind retrieval
- Write to ISIS

Figure 10. FNMOC Scatterometer Processing Flow Diagram.

1. Time and Duplicate Checks

Processing of the data begins after the main routine reads in the data file. The first subroutine, ROWCEL, Figure 10, assigns row and cell numbers to all measurement points. In the standard FDP format each 500 km x 500 km block of data is divided into 19 swaths with 19 measurements across each swath (Figure B.1, Appendix B). The swaths are assigned row numbers (1-19) and the points across the swath are assigned cell numbers (1-19). Each cell across the swath corresponds to a specific range in antenna inclination angle. The subroutine compares the measured inclination angle of the forward antenna to expected ranges for all of the cells, specified in Table 3 and shown in Figure B.1, Appendix B. Measurements are taken from the inner part of the swath to the outer part. Therefore, as the cell numbers exceed 19 the row number is incrementally increased. The row and cell numbers for each measurement point are added as fields to each record and the entire array is returned to the main routine. Following this, records can be filtered without losing the relative spatial location of those remaining.

The current operations time is determined by running a local utility called DTGOPS, which returns the current date-time-group (DTG) in 12 hour increments (00Z or 12Z). The operational DTG is later used to extract ice and wind data from the ISIS database. Because the global model (NOGAPS) runs every six hours, see Figure 11, the DTG might need modification from either 00Z to 06Z or 12Z to 18Z prior to use for data extraction. The time of the first record in the file is used to determine if modification is necessary.
Once the appropriate operational DTG is determined, it is passed to the subroutine WINDOW. This subroutine compares each data point to a time window ± 3 hours from the operational DTG, see Figure 11. A time check is necessary because the calculated scatterometer wind vectors are compared to model wind data for a specific time. If the time difference between the two is too great, the comparison will not be valid. If a record is within the time window it is saved in the data array, otherwise it is saved in a temporary

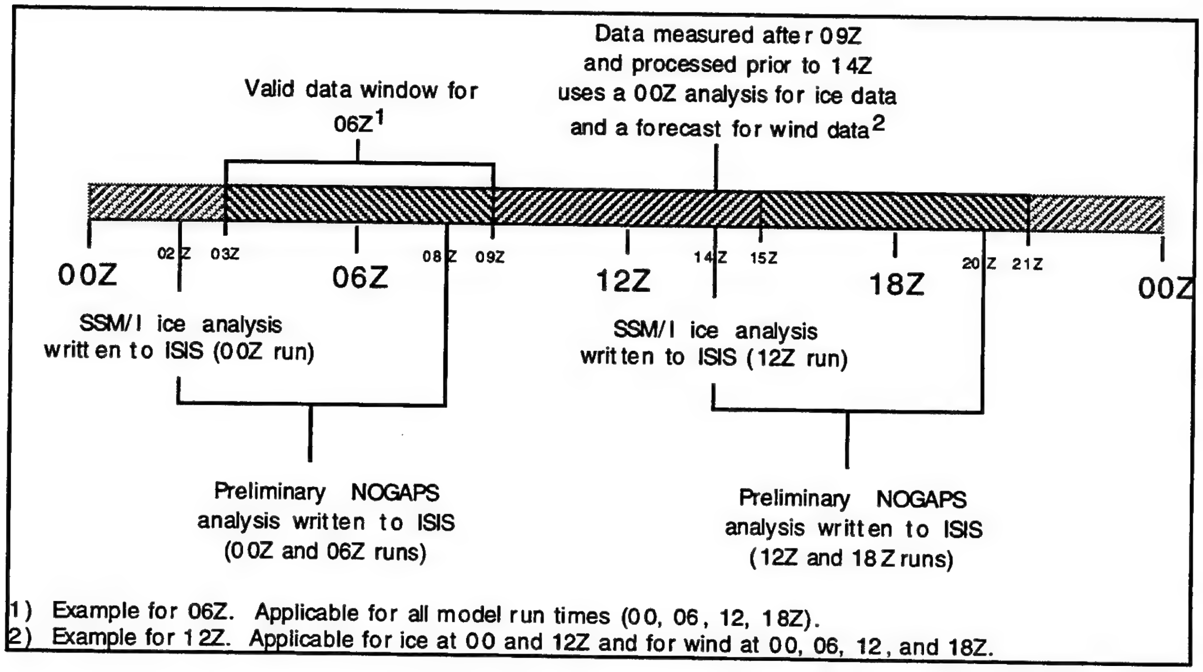
Table 3. Cell Number with respect to Incidence Angle Across the Swath.

Cell #	Inclination Angle (deg)
1	22.4
2	26.0
3	28.4
4	30.7
5	32.9
6	35.05
7	37.1
8	39.05
9	40.9
10	42.7
11	44.45
12	46.1
13	47.7
14	49.2
15	50.65
16	52.1
17	53.45
18	54.7
19	55.9
	58.0

array for later processing and a flag is set. When all data within the current window are processed, the flag is checked and any saved data are then processed. All records in each block have the same measurement time. The time of each block in the data file is saved and is passed to the subroutine TMSORT.

TMSORT sorts the data by time of measurement and checks the blocks for duplication. The subroutine begins by assigning all blocks a consecutive index number. In order of their initial index, the block times are checked to ensure they are sequential. If one is out of order, it is compared to all others to see where it should be relative to following blocks, and all indices are then updated accordingly. This continues until all blocks are checked.

The data are then written to an intermediate 3-dimensional array, dimensioned by the number of records, the number of fields, and the index number. The blocks are then in sequential order. Next, a block duplicity check is performed by taking the time of each block and comparing it with the previous one. If the difference is less than 30 seconds, the block is considered a duplicate and its index is set to a bogus value of zero. To avoid "holes" in the sequential indices (where a duplicate block had been), the blocks are assigned new indices that are incremented by one only if the block is valid. The data is then returned to WINDOW and written back to a 2-dimensional array using the new block indices beginning with one, leaving behind the duplicate blocks of zero index. The index numbers are assigned as a new field to each record and the data array is returned to the main routine.



Valid data window for 06Z[1]
Data measured after 09Z and processed prior to 14Z uses a 00Z analysis for ice data and a forecast for wind data[2]
00Z
02Z
03Z
06Z
08Z
09Z
12Z
14Z
15Z
18Z
20Z
21Z
00Z
SSM/I ice analysis written to ISIS (00Z run)
SSM/I ice analysis written to ISIS (12Z run)
Preliminary NOGAPS analysis written to ISIS (00Z and 06Z runs)
Preliminary NOGAPS analysis written to ISIS (12Z and 18Z runs)
1) Example for 06Z. Applicable for all model run times (00, 06, 12, 18Z).
2) Example for 12Z. Applicable for ice at 00 and 12Z and for wind at 00, 06, 12, and 18Z.

Figure 11. FNMOC Models and Data Timeline.

2. σ° Quality Checks

Quality checks for the individual measurements begin with the subroutine QCHECK, called next by ERS1SCAT. This subroutine checks the data against user selected standards and for "missing" values assigned as -32768 by the decoder. Next, the subroutine CONFID is called to check the confidence flag contained in the FDP (Table 2) for the status of bits relating to the integrity of the measurement, including: fore, mid, and aft beam σ° presence; fore, mid, and aft beam arcing; noise figure greater than 20%; land contamination; checksum error detection; and flags present. The filtered array is returned to QCHECK and the antenna noise figures subsequently checked against a limit of 10%. Finally, if the total number of missing pulses from the three antennas exceeds 15 (out of 32 for each antenna), the integrity of the σ° values is questionable and the record is filtered. The filtered array is returned to the main routine.

3. Land/Ice Checks

The data file is next checked for measurements collected over land by the subroutine LANDMSK using an FNMOC utility called LLSEA, which uses the World Vector Shoreline Plus tables as its reference for the shoreline. LLSEA provides performance at different resolutions. A resolution of 1000 m is used in the scatterometer application because of the accuracy with which the latitudes and longitudes are given.

The measurement points are then checked for ice contamination. The points are compared to a gridded analysis of SSM/I ice measurements, verified by a sea surface temperature analysis. The ice analysis is produced at FNMOC every 12 hours from the 00Z and 12Z model runs, and the data are written to the ISIS database approximately two hours after the runs begin, as shown in Figure 11. The analysis provides percentage of ice concentration for the southern hemisphere (90°S to 54°S) and northern hemisphere (35°N to 90°N). The fields are extracted from the ISIS database either from the current DTG or up to four previous DTGs (-48 hours) if there are no data available. If the fifth extraction attempt is unsuccessful , processing will be halted and an error message generated.

Once the hemispheric ice data are extracted, the x and y locations of the scatterometer points are passed to the OMSP routine FINTRP which interpolates the ice values to these locations. Any scatterometer x/y coordinates outside the grid limits are assigned a bogus value ($1x10^{10}$). All values corresponding to scatterometer measurement points between 54°S and 35°N are set to zero, representing no ice. Finally, the ice array is compared to the corresponding scatterometer records and any point with greater than a 55% ice concentration is eliminated. Long-term studies by the ECMWF revealed an ice concentration of 55% was the most appropriate to fix the sea ice limit [Ref. 16].

4. NOGAPS Wind Field Collocation

At this point, it is expected that all data remaining will be processed into wind vectors. Therefore, it is practical to then retrieve corresponding wind data produced by the NOGAPS model. The NOGAPS global 10m sea-surface wind field is a gridded field available as an analysis and forecast from each model run (00Z, 06Z, 12Z, and 18Z) and is available in ISIS approximately two hours from the start of the run (see Figure 11). Collocation is performed by the subroutine WINDDATA, having been passed arrays of the scatterometer latitudes and longitudes and the modified operational DTG. Similar to the ice data extraction, the same OMSP routines are used to convert the scatterometer latitudes and longitudes to points on the 360 x 181 global wind grid. The wind data is available in polar coordinates, by speed and direction, and in Cartesian coordinates, by "u" and "v" vector components. The u and v components* are extracted because it is more accurate to

* The u and v Cartesian coordinates for the collocated NOGAPS wind vectors are converted to polar coordinates using an local utility which returns wind directions in the oceanographic "direction toward" convention (ranging from -180° to 180°). The scatterometer wind vectors produced are in the meteorological "direction from" convention (ranging from 0° to 360°). The NOGAPS vectors are converted to the meteorological convention by adding 180°.

interpolate the vector components rather than the vector's magnitude and direction.

The first extraction looks for the analysis from the most recent model run using the operational DTG passed to the subroutine. If the extraction is unsuccessful, the subroutine begins looking for forecasts for that DTG. This is accomplished by incrementing the DTG back six hours and setting the forecast period to six. For example, if the most recent model run was at a DTG of 1996010112 (i.e., 1200, 1 January 1996) and an analysis for that DTG is not available, the DTG would be changed to 1996010106 (i.e., 0600, 1 January 1996) and the subroutine would request a six hour forecast. If the 06Z forecast is not available from ISIS, the subroutine will continue to look back in time for longer forecast periods until the forecast period is 24. No available NOGAPS wind data is considered a fatal error. An error message will then be generated and processing will halt.

5. Continuity Checks

ERS1SCAT calls CONTIN next to determine which of the filtered blocks of data are continuous, i.e., within 1.5 minutes of each other. A flag value is set to 1 if the block is continuous with the previous block. A flag value of 0 indicates the first block in a continuous set. The flag values are used with the subroutine SLICE during wind retrieval. The number of records remaining in each block are also saved to an array and, along with the continuity flags, are returned to the main routine.

6. Wind Retrieval

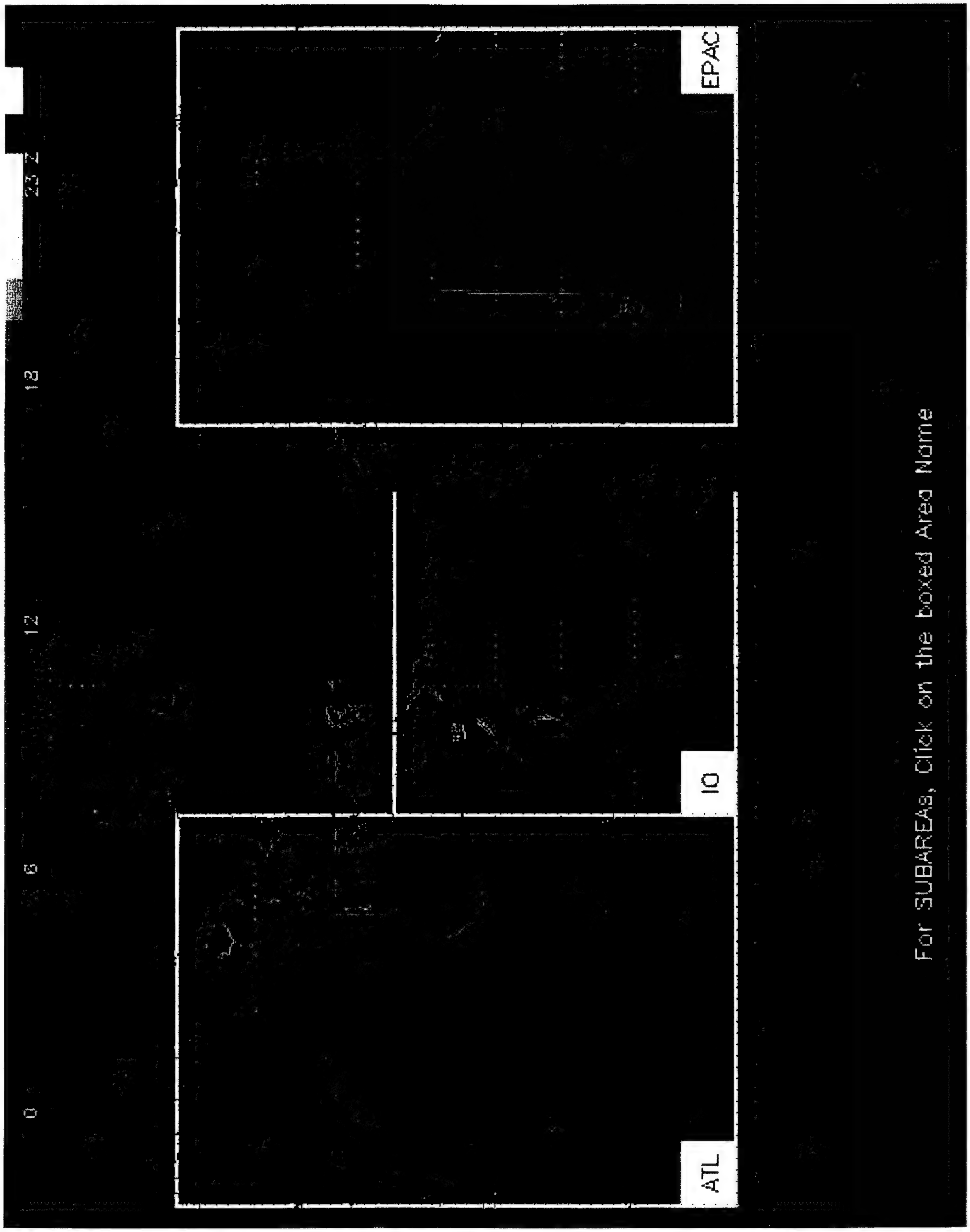
Wind retrieval is performed in the same way it is performed at NCEP, discussed in detail in Chapter III, Section D and shown in Figure 9. WINRET produces first guess solutions (up to four) for the scatterometer measurements. The subroutine EXTFIT then compares the first guess solutions to the NOGAPS background field and ranks the solutions according to their relative fit to the NOGAPS field. To avoid discontinuities in the final step, SLICE, dummy data arrays for each field type (361 records per block) are filled with bogus values and then written over with the real data. Next, contiguous blocks of arrays are arranged using the continuity flag for each block, where a flag of 0 begins a new set of contiguous blocks. The contiguous blocks and the solutions and probabilities are then passed to the subroutine SLICE. When SLICE is finished the records are written to the ISIS database. The code then returns to get the next set of contiguous blocks of data and begins the SLICE process again until all blocks are checked.

The FDP data and the ranked solutions are written to the ISIS database as structures, referred to by the dataset name "sctr," with the subroutine WRITE_LLT (Latitude/Longitude/Time format). This allows subsequent extraction from the database using either space or time. When all data have been written to ISIS, PROCESS calls the ISIS utility LCLOS to close the scatterometer table in the database.

D. PRODUCTS

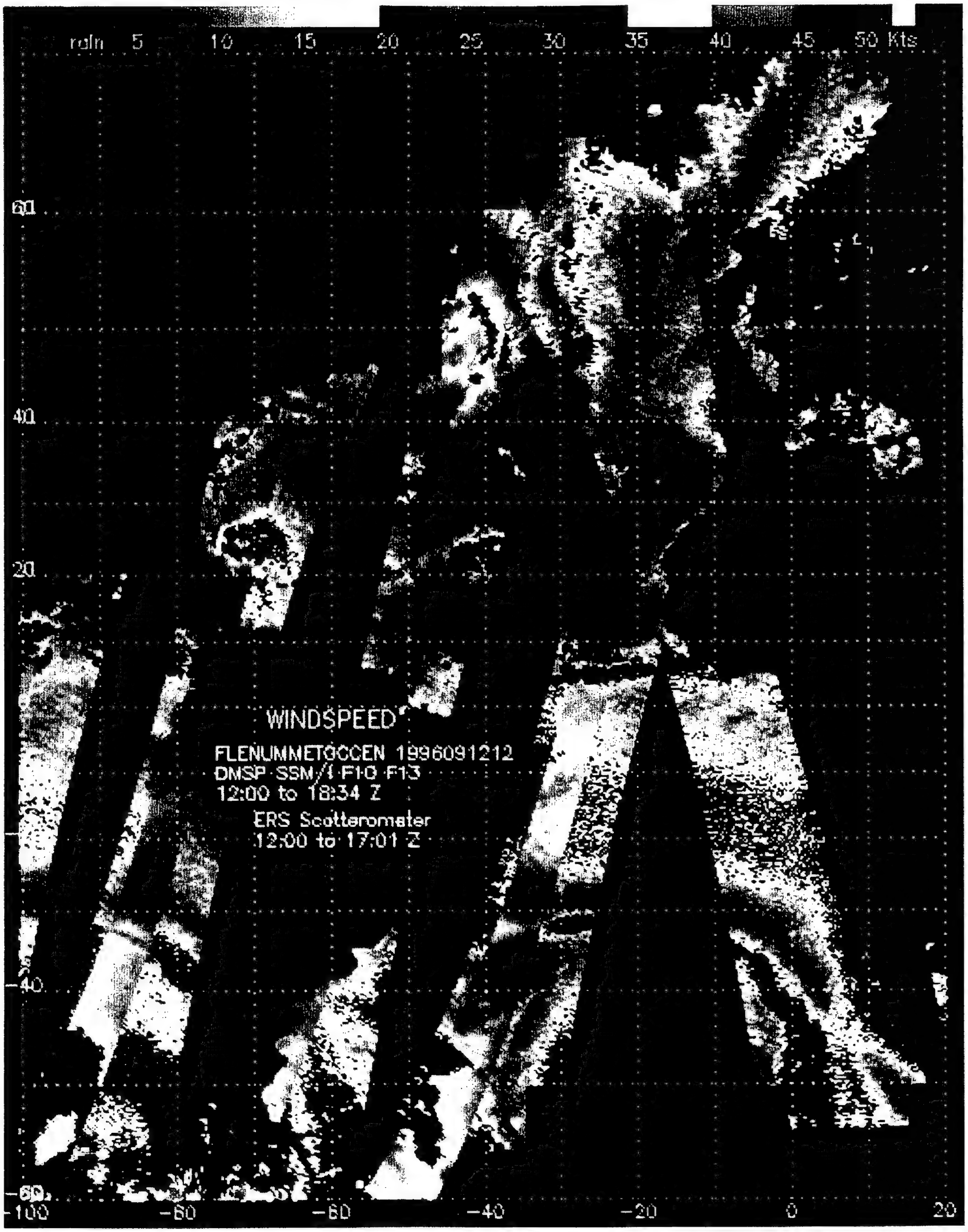
Once the processed scatterometer data are written to the ISIS database, there are two primary applications: 1) extraction of wind vectors for assimilation in NOGAPS and other FNMOC models, and 2) extraction of vectors for near real-time graphics for distribution to other Navy regional centers via FNMOC's Joint METOC Viewer (JMV). The latter will replace scatterometer graphics currently produced by the Naval Oceanographic Office for internet access. Work for assimilation of scatterometer vectors into FNMOC models will be performed by the Naval Reasearch Laboratory, Monterey, California, and has not yet begun. The graphics product, created by FNMOC personnel, is in development and will provide scatterometer wind vectors overlaying Defense Meteorological Satellite Program SSM/I wind speeds.

The graphics will be available for Navy users and will provide global data in pictures of increasing scale and detail. Figure 12 shows the initial JMV screen which provides a global view of the data available from the previous 11 hours (the actual product will provide the most recent six hours), color-coded according to the time of measurement. The global map is broken down into areas of responsibility (AOR). The enhanced view of each AOR is opened when the user selects the name of the desired area. The Atlantic AOR is provided as an example in Figure 13. The AOR graphics provide wind speed only, which is specified by the color table along the top of the picture. In all available pictures, the scatterometer can be distinguished by its smaller swath width. From the AOR maps, ten degree boxes can be selected to display the most enhanced view, providing scatterometer wind barbs which indicate both wind speed and direction. SSM/I measurement points are indicated by numbers representing the magnitude of the wind speed at that location. Both data types are again color-coded according to speed. The product also allows for viewing of either scatterometer or SSM/I data. Typical resolution of the ten degree boxes is shown in Figure 14.



For SUBAREAs, Click on the boxed Area Name

Figure 12. Initial JMV Screen for Scatterometer/SSMI Product.



FLENUMMETOCCEN 1996091212
DMSP SSM/I F10 F13
12:00 to 18:34 Z
ERS Scatterometer
12:00 to 17:01 Z

Figure 13. JMV Atlantic AOR Screen for Scatterometer/SSMI Product.

V. RESULTS AND CONCLUSIONS

A. VALIDATION

The European Space Agency's scatterometer Fast Delivery Product is frequently deficient, particularly in wind direction. Therefore, the method outlined in Chapter IV was implemented at the Fleet Numerical Meteorology and Oceanography Center to process raw scatterometer data and produce improved wind vectors. Output data from the FNMOC processing method are validated by comparison with the ESA Fast Delivery Product. Though it is of primary importance to demonstrate an improvement over the FDP, much can be gleaned about the nature of scatterometer data from a comparison with other sources of ocean surface wind information, such as the FNMOC global model surface wind field, the Defense Meteorological Satellite Program SSM/I wind speeds, and aircraft or ship measurements. Comparison of data is performed for the specific case of hurricane Hortense (4-15 September 1996, sustained winds of 120 mph).

The swath over hurricane Hortense was taken on 12 September 1995 at 1514 by the ERS-2 scatterometer (Figure 14), when the hurricane eye was approximately located at 24° N and 72° W. The Defense Meteorological Satellite Program F10 satellite took SSM/I measurements over the same area at 1515 on the same day, providing an excellent comparison between the two instruments.

1. ESA Fast Delivery Product

The scatterometer Fast Delivery Product provided by the European Space Agency often renders wind fields that are not accurate, such as neighboring wind vectors that directly oppose each other (i.e., vectors are 180° apart) and wind vectors that do not follow other general patterns of known meteorological phenomena. In the case of hurricane Hortense, the FDP (Figure 15) displays such inconsistencies as opposing

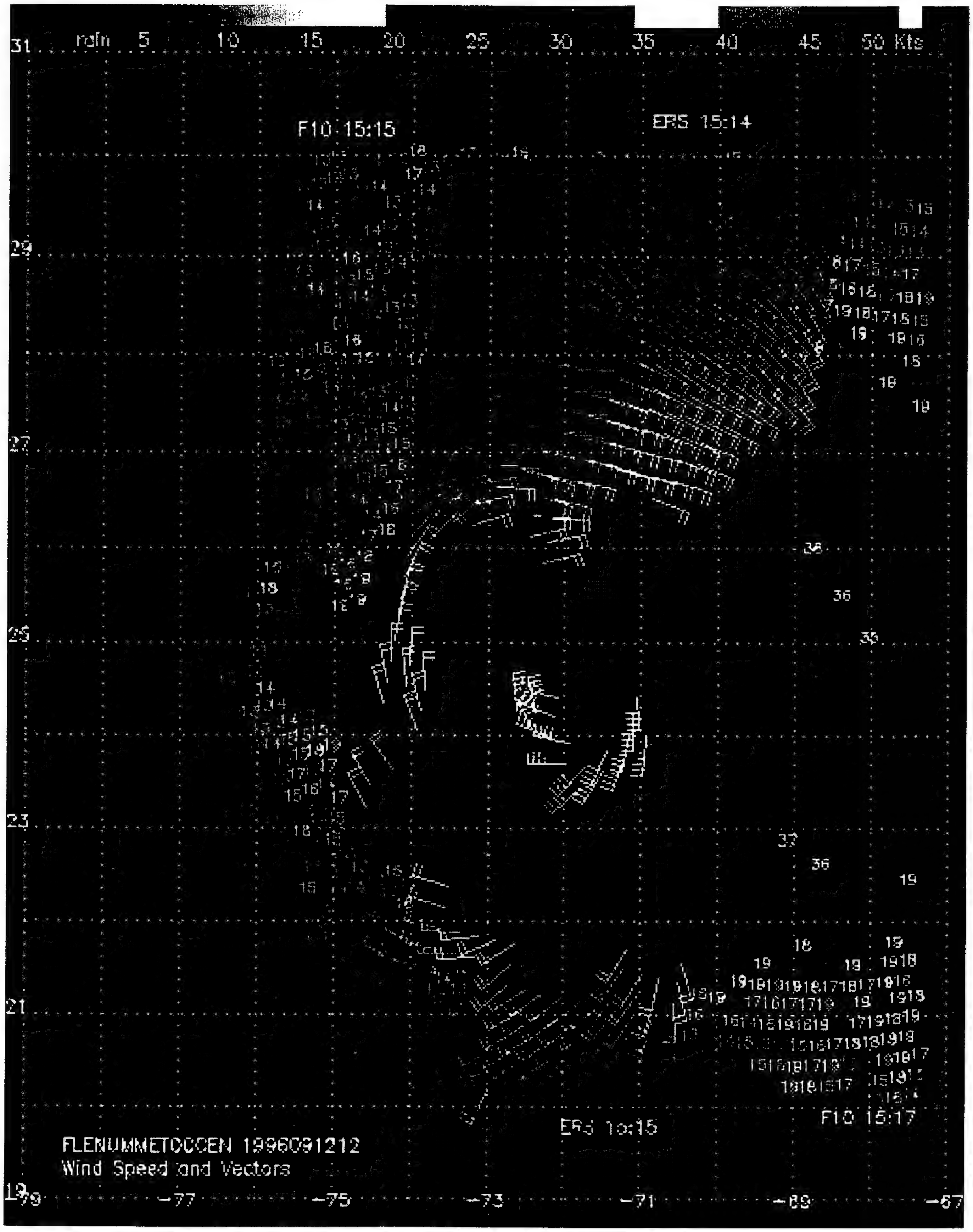
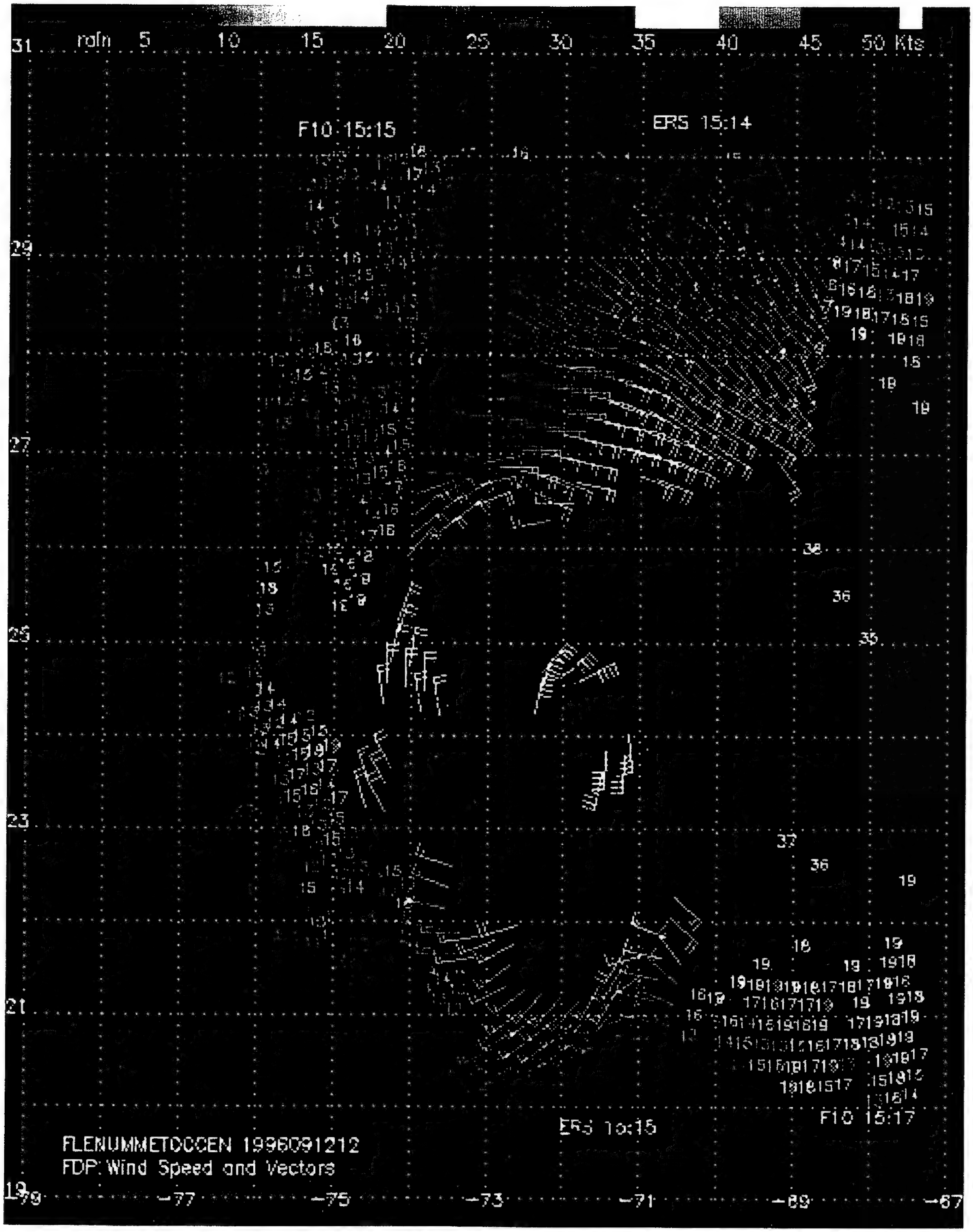


Figure 14. FNMOC-Produced Scatterometer Data for Hurricane Hortense.



31
m/h 5 10 15 20 25 30 35 40 45 50 Kts
F10 15:15
ERS 15:14
29
27
25
23
21
ERS 15:15
F10 15:17
FLENUMMETOCCEN 1996091212
FDP: Wind Speed and Vectors
19
-78 -77 -75 -73 -71 -69 -67

Figure 15. FDP Scatterometer Data for Hurricane Hortense.

vectors at positions 26° N x 72° W, 22°N x 71° W, and, more importantly, over the eye of the hurricane. This wind field, therefore, shows the concentrations of highest wind speed on the north-west and south-east sides of the eye. The exact location of the eye is difficult to discern because of the opposing vectors. The eye can be approximately fixed at 24° N and 72° W.

The FNMOC (NOGAPS-adjusted) scatterometer vectors (Figure 14) show a more smooth flow, i.e., vectors follow a consistent pattern. The correction by the NOGAPS comparison has also adjusted the vectors around the eye, resulting in the expected counter-clockwise flow and higher wind speeds (maximum of approximately 50 knots) on the south side of the eye. There is also a resulting adjustment, though slight, of the location of the eye to approximately 24.5° N and 71.9° W.

2. FNMOC Global Model, NOGAPS

The scatterometer wind field produced at FNMOC is highly influenced by the NOGAPS global model. Comparison with the model's 10m sea-surface wind field is the primary means of ambiguity removal in processing the scatterometer data into wind vectors. As assimilation of the scatterometer data into the model has not yet begun, the model has not been influenced by the data and, therefore, the difference between the two is greatest in this situation. Having said this, the NOGAPS field used for ambiguity removal of the scatterometer wind vectors over hurricane Hortense (Figure 14) is shown, collocated to the scatterometer points, in Figure 16.

The NOGAPS plot shows a smooth counter-clockwise flow around the eye of the hurricane. The concentration of highest winds is on the north-east side of the eye, consistent with analyses from the ship and aircraft measurements (Figure 18 and Figure 19) covering the same time period. Though the apparent accuracy of the model should

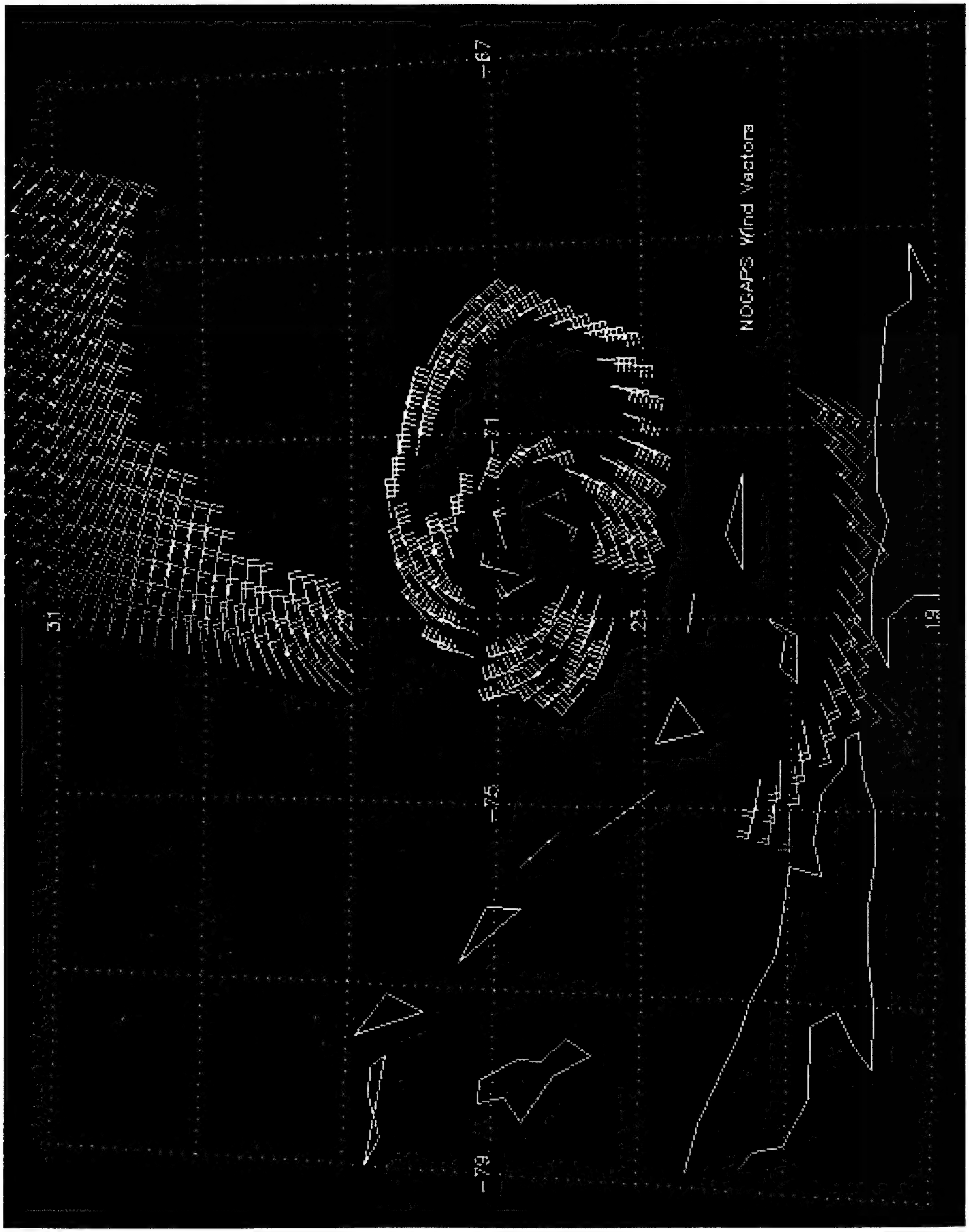


Figure 16. NOGAPS Wind Data for Hurricane Hortense.

favorably affect the scatterometer solutions, the eye of the hurricane in the NOGAPS field is translated approximately 0.5° to the west (24.5° N and 72.3° W) from the location of the eye in the scatterometer field. Therefore, optimal point by point ambiguity removal was not performed. This is an inherent problem in using a numerical model for ambiguity removal over distinctive weather patterns, such as tropical storms and fronts. It should also be pointed out that the model, similar to the satellite measurements, shows maximum wind speeds of 50 knots.

3. FNMOC SSM/I

The DMSP SSM/I is a passive microwave radiometer that takes measurements at frequencies of 19 and 22 GHz for wind *speed*, as opposed to the scatterometer which operates at 5.3 GHz for wind speed and *direction*. The DMSP F10 satellite has nearly the same orbit as ERS-2 and, therefore, provides a means to compare measurements taken by each over the same location at the same time. Comparison with the SSM/I also allows the opportunity to compare atmospheric effects on the measurements from each instrument. Although data from both instruments are affected by rain, the frequency at which the SSM/I operates is more susceptible to rain contamination, which is most severe in areas of extreme weather conditions such as hurricanes. Rain contamination can be seen as purple dots or, for extreme contamination, as missing measurements in SSM/I figures in Appendix E. A swath by F10 over hurricane Hortense (Figure 17) shows significant rain contamination in the area of the hurricane. The contamination is so significant in the area of the eye that its location and the area of highest wind are unavailable. This is particularly undesirable as it is areas of extreme weather where information is most necessary for naval operations.

The measurements taken by the scatterometer, at the lower frequency, provide information over most of the hurricane. Therefore, the scatterometer data are a perfect

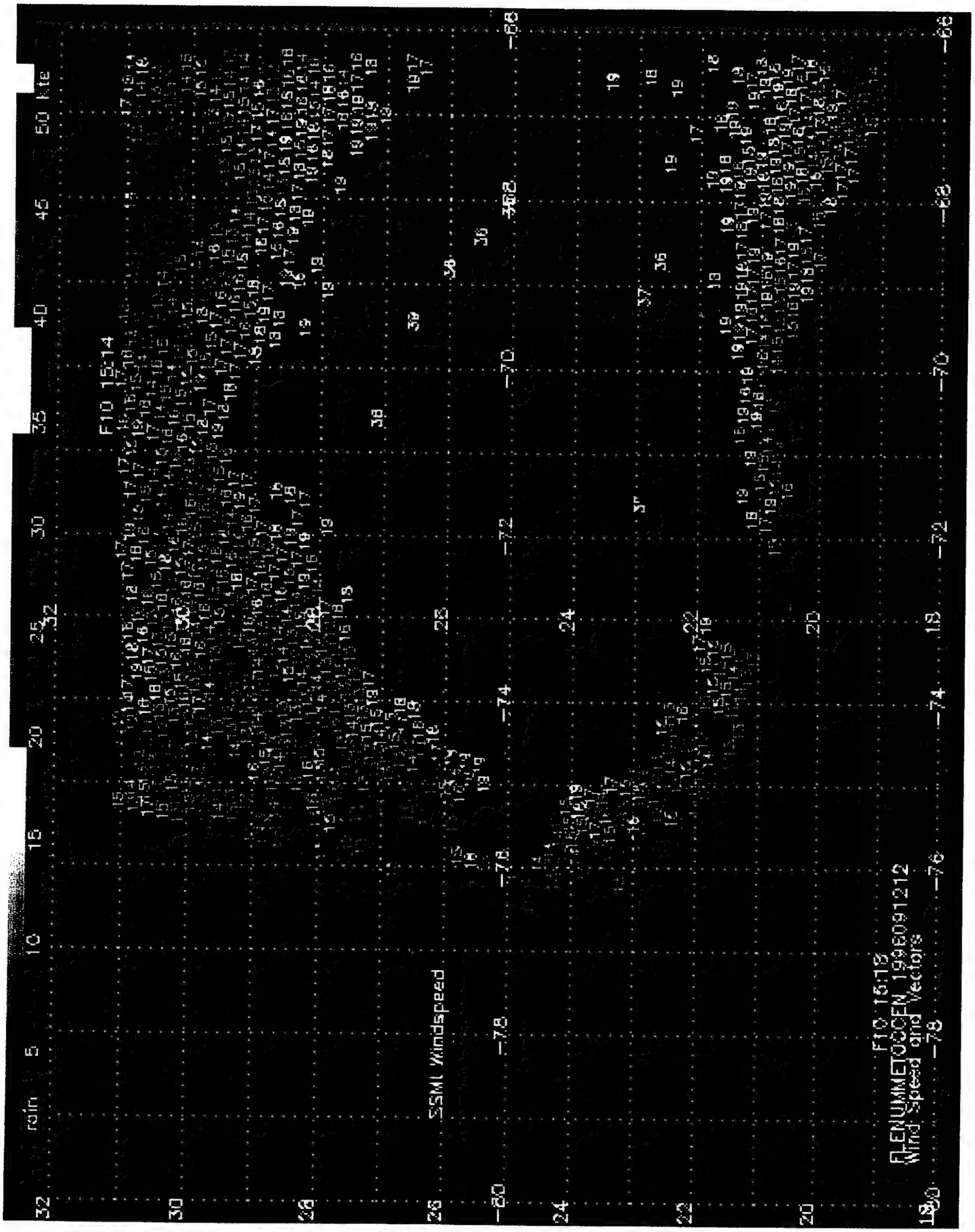


Figure 17. SSM/I Data for Hurricane Hortense.

complement to the SSM/I wind speed measurements. Comparing the two sets of measurements, it can be seen that the SSM/I consistently displays slightly higher wind speeds than the scatterometer. The difference could be due to the atmospheric water concentration and the effect on the high frequency measurements of the SSM/I. In general, the two data types behave in a very similar fashion, producing similar pictures of the wind field.

4. In-Situ Measurements

It is worthwhile to compare the scatterometer wind vectors with available in-situ measurements to observe the general behavior of the data taken by the scatterometer. In-situ measurements can be provided by aircraft, ships, or buoys. For hurricane Hortense the Hurricane Research Division of NOAA provided an analysis from aircraft and ship measurements for 1300 and 1930 (Figure 18 and Figure 19, respectively). Comparing the two, it is observed that the hurricane is traveling essentially due north at 71.8° W from 24° N to 24.8° N. The concentration of maximum winds also shifts from 90 knots north-east of the eye to 95 knots south-east of the eye. The latter of the two (Figure 19) shows very similar behavior to the FNMOC-produced wind field (Figure 14) when comparing the analysis isotachs (contours of speed) and the contours of the colored regions of the scatterometer wind field. Though the general behavior of the fields are very similar, it is immediately obvious the magnitude of the speed of the scatterometer field is almost half of the in-situ measurements.

As an additional comparison, an image from the National Oceanic and Atmospheric Administration (NOAA) Geostationary Operational Environmental Satellite (GOES) is provided in Figure 20. This image (visible spectrum) of Hortense was taken at exactly the same time as the scatterometer measurements, 12 September 1995 at 1515, and shows the

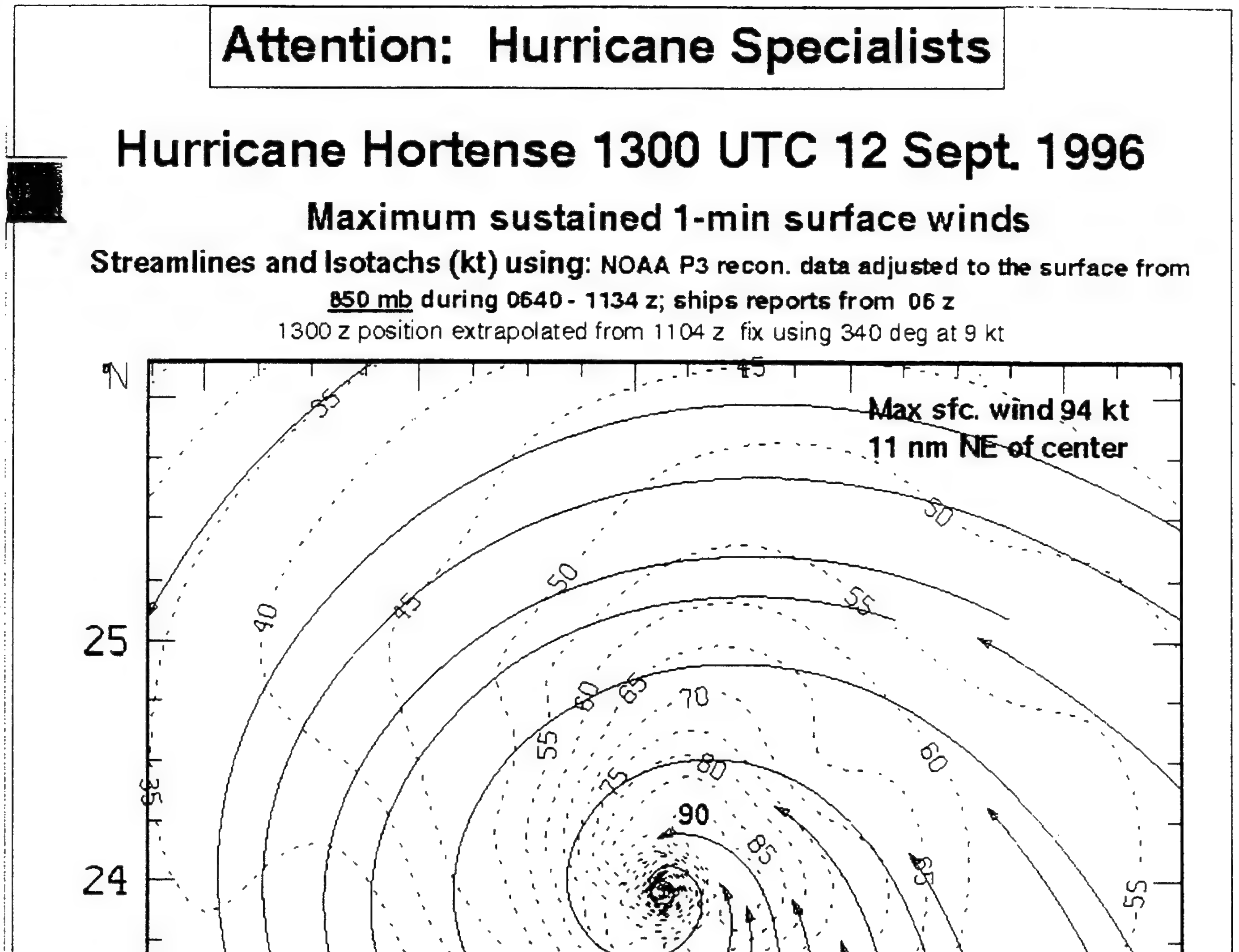

Attention: Hurricane Specialists
Hurricane Hortense 1300 UTC 12 Sept. 1996
Maximum sustained 1-min surface winds
Streamlines and Isotachs (kt) using: NOAA P3 recon. data adjusted to the surface from 850 mb during 0640 - 1134 z; ships reports from 06 z
1300 z position extrapolated from 1104 z fix using 340 deg at 9 kt
°N
25
24
23
22
74
73
72
71
70
°W
Max sfc. wind 94 kt
11 nm NE of center
90
NOT VALID OVER LAND
Valid only for MARINE exposure at 10 m
Experimental research product of :
NOAA / AOML / Hurricane Research Division

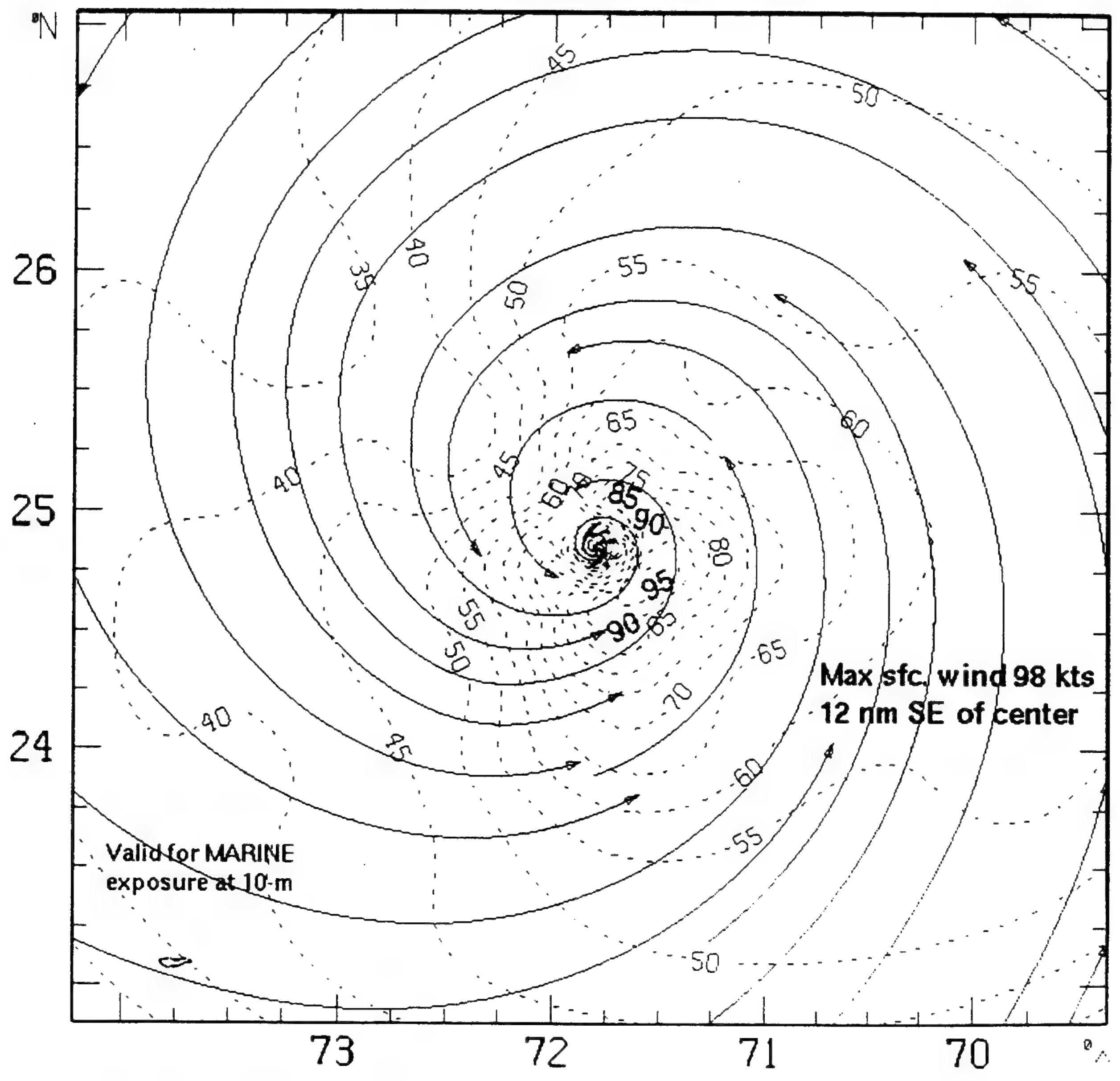
Figure 18. In-Situ Analysis for Hurricane Hortense at 1300Z.

Attention Hurricane Specialists

Hurricane Hortense 1930 UTC 12 Sept. 1996

Maximum sustained 1-min surface winds

Streamlines and Isotachs (kt) using: U. S. Air Force Reserves recon. data adjusted to the surface from 700 mb during 1648-1919 z, ships from 15-18Z
1930 Z position extrapolated from 1856 Z fix using 340 deg at 9 kts



Experimental research product of :
NOAA / AOML / Hurricane Research Division

Figure 19. In-Situ Analysis for Hurricane Hortense at 1930Z.

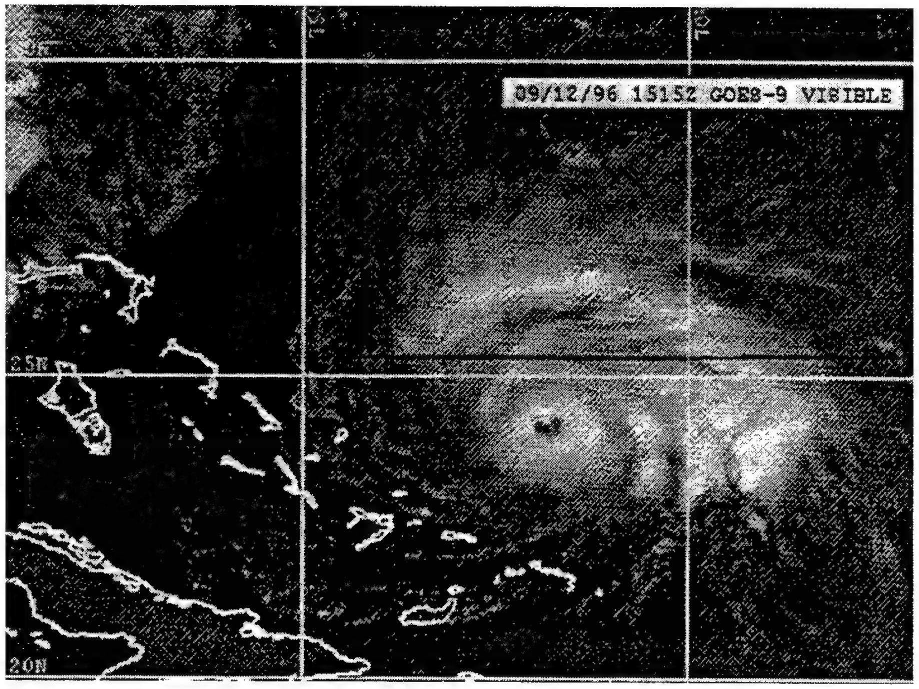


Figure 20. GOES Visual Image of Hurricane Hortense at 1515.

location of the eye at approximately 24.5° N and 71.8° W, in corroboration with the scatterometer position.

5. Performance Statistics

The scatterometer data are received at FNMOC from ESA via NOAA. Once the data are at FNMOC, the files are decoded and processed for use by the local models and in the graphics product. The data are available to FNMOC, on average, within the FDP specification of three hours. In the operational mode, each file will be decoded as it is received. The decoding process takes less than one minute, after which it is moved to the Cray J90 where it will be immediately processed. File size ranges from a minimum of approximately 0.2 Mb to a maximum of 4 Mb. The average file size is 1.8 Mb. Table 3 provides examples of processing time and the number of 500 x 500 km blocks contained in files of 0.2, 1.8, and 4 Mb.

Table 4. File Processing Time Statistics

File size (Mb)	Blocks contained	Processing time (sec)
0.2	3	38
1.8	29	44
4	61	59

Once the data are processed they are immediately written to the FNMOC database and are available for use. The model will use the data every six hours coincident with the operational model run times. The graphics product is expected to be produced every hour and contain the most recent seven hours of data. Currently production of the product takes approximately 45 minutes. Changes in computer platforms may improve processing time for the product, and the schedule for production may be adjusted according to feedback from operational users.

B. CONCLUSIONS

Various comparisons show a consistent, and sometimes dramatic, improvement in the FNMOC-produced scatterometer wind vectors over the ESA Fast Delivery Product, therefore validating the method and the time it takes to create the solutions. The primary correction observed is in wind direction, with the FNMOC wind vectors often showing a more consistent and realistic field. Hurricane Hortense provided an excellent opportunity to compare the behavior of the FNMOC-produced wind vectors with the FDP and to compare scatterometer data in general with other sources of wind data for an extreme, and therefore important, situation. Location of the eye and the maximum wind speed are two quantitative characteristics of the hurricane that provide a means of comparison. A summary of values for hurricane Hortense for the compared data types is provided in Table 5.

Differences in direction of the FNMOC-produced wind vectors from the FDP also result in different solutions for wind speed. In the case of hurricane Hortense, by correcting the opposing winds across the eye in the FDP, the FNMOC vectors had a higher maximum speed of 50 knots over a larger area. The FNMOC wind field also shows a single concentration of high wind speeds, whereas the FDP shows two areas of highest wind speed. A single area of high wind speed around the eye of a hurricane is consistent with historical knowledge and with the in-situ measurements taken of Hortense (Figure 18 and Figure 19).

Table 5. Comparison of Eye Location and Maximum Wind Speed for Hurricane Hortense Between Various Data Types

Data Source	Max Wind Speed (knts)	Location of Eye: Latitude	Longitude
FNMOC Scatterometer	50	24.5° N	71.9° W
FDP Scatterometer	50	24° N	72° W
FNMOC NOGAPS	50	24.5° N	72.3° W
DMSP F10 SSM/I	-	-	-
Aircraft and Ship Analyses	90-95	24°-24.8° N	71.8° W
GOES-9	-	24.5° N	71.8° W

In terms of wind flow behavior, the scatterometer data appears to be accurate in most situations. This was demonstrated by the GOES-9 image of hurricane Hortense which was in agreement with the scatterometer wind field as to the location of the eye, whereas the NOGAPS model placed the eye incorrectly. However, it is clear in situations of extreme weather that the scatterometer wind speed magnitudes are not representative of the actual situation. For example, the maximum speed shown by the scatterometer data is 50 knots, while the analyses from the in-situ aircraft and ship measurements show maximum speeds of 90 to 95 knots. It should be restated that the current model function, CMOD4, has inherent deficiencies which affect the ability to return high wind speeds. The model function is considered to be valid only in the range of 4-18 m/s (7.8-35 knots), since higher wind speeds were not contained in the calibration data set. Also, additional problems with returning high wind speeds are due to effects by geophysical parameters other than wind, such as rain or sea state. Additionally, comparisons with high winds show the required backscatter values are too high to be recovered correctly from CMOD4.

The space-borne scatterometer is currently the only source of global sea-surface wind speed and direction. The few examples and comparisons provided here show the wind vectors produced by scatterometer data can provide an accurate picture of the ocean wind field. These examples also demonstrate, however, an inability to produce high wind speeds representative of the true situation. Even so, the scatterometer supplies accurate ocean wind field information on a scope never before available. The ESA Fast Delivery Product is the fastest means to receive this data, however it often lacks the degree of accuracy the scatterometer has the potential to provide. The processing method implemented at FNMOC provides improved wind vectors and a much more accurate wind field, such that the vectors are available for use within minutes of receipt by FNMOC. Until the FDP is improved, the FNMOC processing method is necessary to provide the most accurate data to the U.S. Navy fleet.

C. FUTURE WORK

Intensive studies performed at NASA's Goddard Space Flight Center have shown scatterometer data has a substantial impact on numerical models, particularly in the Southern Hemisphere where there are few sources of wind data. It is therefore of primary importance that FNMOC begin assimilating this data into its local models, primarily NOGAPS. Once the data is assimilated into the model on an operational basis, the difference between the NOGAPS wind field and that produced from scatterometer measurements should decrease and the ambiguity removal process improve. Even so, a difference in the location of characteristic weather conditions, such as fronts or tropical storms, between the model and the scattermeter measurements can present a significant problem in ambiguity removal. The example of hurricane Hortense demonstrated this to a small degree, as the model placed eye of the hurricane approximately 0.5° from the actual location shown in the GOES-9 image. If, on the other hand, the eye or front is displaced a degree or more between the model and the scatterometer data, the ambiguity removal process can produce solutions as much as 180° from the true situation, producing an absurd representation of the actual wind field. Investigation into this phenomenon and methods to resolve its consequences should be performed and the methods implemented to yield an optimal scatterometer product.

The next significant step in scatterometery has already taken place with the launch of the NASA scatterometer (NSCAT) on the Japanese Advanced Earth Observing Satellite (ADEOS). There is much to be done to achieve efficient delivery of accurate global NSCAT wind data to operational centers, such as algorithm refinement, determination of the data flow path and ground processing requirements, data validation, and implementation of data processing. When NSCAT transitions into its operational mode there will, for the first time, be two simultaneous sources of scatterometer data, providing unprecedented global coverage. As the NSCAT operates at a frequency of 13.9 GHz and the ERS scatterometer has an operating frequency of 5.3 GHz, it will be useful to perform studies comparing the accuracy of data taken at the different frequencies and the severity of atmospheric effects on each. Finally, it is absolutely necessary to continue to take advantage of this unique source of data, particularly as more becomes available, to improve the accuracy of numerical modeling and to provide the most accurate, useful data to operational centers around the world.

APPENDIX A. THE CMOD4 MODEL TRANSFER FUNCTION

The CMOD4 model function developed by Stoffelen [Ref. 10] is the version currently in use at the European Space Agency for the scatterometer Fast Delivery Product, and is one chosen for use at the National Center for Environmental Prediction and the Fleet Numerical Meteorology and Oceanography Center. It is of the form:

$$\sigma^\circ_{lin} = b_0(1 + b_1 \cos\phi + b_3 \tanh b_2 \cos 2\phi)^{1.6}$$

where $b_0 = b_r 10^{\alpha+\gamma f_1(V+\beta)}$

and $f_1(y) = \begin{cases} 0 & ,y \le 0 \\ \log y & ,0 < y \le 5 \\ \sqrt{y}/3.2 & ,y > 5 \end{cases}$

and α, β, γ, b_1, b_2, and b_3 are expanded as Legendre polynomials to a total of 18 coefficients, see Table A1. b_r is a residual correction factor to b_0 and is given as a look-up table, Table A2, as a function of incidence angle:

$$\alpha = c_1P_0 + c_2P_1 + c_3P_2$$
$$\gamma = c_4P_0 + c_5P_1 + c_6P_2$$
$$\beta = c_7P_0 + c_8P_1 + c_9P_2$$
$$b_1 = c_{10}P_0 + c_{11}V + (c_{12}P_0 + c_{13}V)f_2(x)$$
$$b_2 = c_{14}P_0 + c_{15}(1 + P_1)V$$
$$b_3 = 0.42(1 + c_{16}(c_{17} + x)(c_{18} + V))$$
$$b_r = LUT(\theta)$$
$$f_2(x) = \tanh[2.5(x + 0.35)] - 0.61(x + 0.35)$$

where the Legendre polynomials are:

$P_0=1$, $P_1=x$, $P_2=(3x^2-1)/2$ with $x=(\theta-40)/25$.

V is wind speed in m/s, ϕ is wind direction relative to the radar look angle, and θ is the incidence angle. Given is the linear form of σ°. To convert to decibels use $\sigma^\circ_{dB} = 10\log\sigma^\circ_{lin}$. [Ref. 10]

Table A.1 CMOD4 Coefficients.

Coefficient		Value
α	c_1	-2.301523
	c_2	-1.632686
	c_3	0.761210
γ	c_4	1.156619
	c_5	0.595955
	c_6	-0.293819
β	c_7	-1.015244
	c_8	0.342175
	c_9	-0.500786
$\mathbf{b_1}$	c_{10}	0.014430
	c_{11}	0.002484
	c_{12}	0.074450
	c_{13}	0.004023
$\mathbf{b_2}$	c_{14}	0.148810
	c_{15}	0.089286
$\mathbf{b_3}$	c_{16}	-0.006667
	c_{17}	3.000000
	c_{18}	-10.00000

Table A.2 Residual Factors for CMOD4.

$\theta°$	b_r	$\theta°$	b_r
16	1.075	39	0.988
17	1.075	40	0.998
18	1.075	41	1.009
19	1.072	42	1.021
20	1.069	43	1.033
21	1.066	44	1.042
22	1.056	45	1.050
23	1.030	46	1.054
24	1.004	47	1.053
25	0.979	48	1.052
26	0.967	49	1.047
27	0.958	50	1.038
28	0.949	51	1.028
29	0.941	52	1.016
30	0.934	53	1.002
31	0.927	54	0.989
32	0.923	55	0.965
33	0.930	56	0.941
34	0.937	57	0.929
35	0.944	58	0.929
36	0.955	59	0.929
37	0.967	60	0.929
38	0.978		

APPENDIX B. WIND SCATTEROMETER FAST DELIVERY PRODUCT

Table B.1 ESA Wind Scatterometer Fast Delivery Product.

Field Number	Description	Units/Reference
1	Satellite identifier: 1 = ERS-1 2 = ERS-2	code table/0
2	Software identifier	number/0
3	Identifieer of message originating center	code table/0
4	Identifier of product ground station	code table/0
5	Satellite track	degrees/0
UTC time of ascending node and state vector		
6	Year	year/0
7	Month	month/0
8	Day	day/0
9	Hour	hour/0
10	Minute	minute/0
11	Second	sec*10^{-3}/0
12	State vector: X location	m*10^{-2}/-1073741824
13	State vector: Y location	m*10^{-2}/-1073741824
14	State vector: Z location	m*10^{-2}/-1073741824
15	State vector: X velocity	m/s*10^{-5}/-1073741824
16	State vector: Y velocity	m/s*10^{-5}/-1073741824
17	State vector: Z velocity	m/s*10^{-5}/-1073741824
18	Satellite instrument (=8, i.e. bit 4 set)	flag table/0
19	Year	year/0
20	Month	month/0
21	Day	day/0
22	Hour	hour/0
23	Minute	minute/0
24	Second	sec*10^{-3}/0
25	Latitude	deg*10^{-2}/-9000
26	Longitude	deg*10^{-2}/-18000
Fore beam data:		
27	Radar incidence angle	deg*10^{-1}/0
28	Radar look angle	deg*10^{-1}/0
29	Backscatter ($\sigma°$)	dB*10^{-2}/-5000
30	Noise figure (Kp)	percent*10^{-1}/0
31	Missing packet counter	number/-127
Mid beam data:		
32	Radar incidence angle	deg*10^{-1}/0
33	Radar look angle	deg*10^{-1}/0
34	Backscatter ($\sigma°$)	dB*10^{-2}/-5000
35	Noise figure (Kp)	percent*10^{-1}/0
36	Missing packet counter	number/-127
Aft beam data:		
38	Radar incidence angle	deg*10^{-1}/0
38	Radar look angle	deg*10^{-1}/0
39	Backscatter ($\sigma°$)	dB*10^{-2}/-5000
40	Noise figure (Kp)	percent*10^{-1}/0
41	Missing packet counter	number/-127
42	Wind speed at 10m	m/s*10^{-1}/0
43	Wind direction at 10m	degrees/0
44	UWI product confidence	flage table (Table B.2)/0

Table B.2 ESA Fast Delivery Product Confidence Flag.

Bit	Meaning when set
1	No forebeam calculation
2	No midbeam calculation
3	No aftbeam calculation
4	Forebeam arcing detected
5	Midbeam arcing detected
6	Aftbeam arcing detected
7	Any Beam Kp above or equal to threshold
8	Land (any land in cell footprint)
9	Autonomous ambiguity removal not used
10	Meteorological background not used
11	Minimum residual exceeded threshold
12	Frame checksum error detected
All 13	Missing value

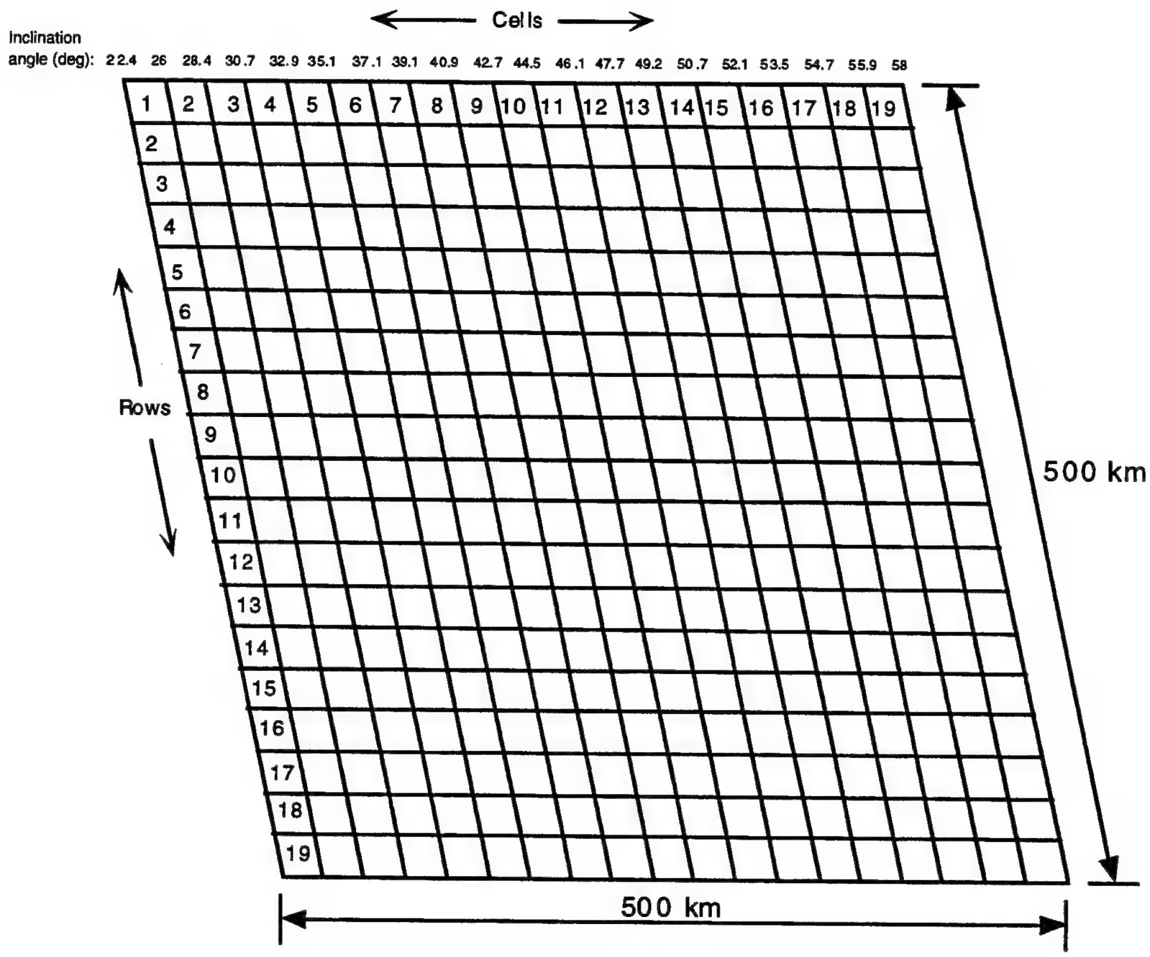

Cells
Inclination angle (deg): 22.4 26 28.4 30.7 32.9 35.1 37.1 39.1 40.9 42.7 44.5 46.1 47.7 49.2 50.7 52.1 53.5 54.7 55.9 58
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
Rows
500 km
500 km

Figure B.1 Scatterometer Data Block

APPENDIX C. EXAMPLE: WIND RETRIEVAL

The processing of scatterometer antenna measurements into wind vectors consists of two primary steps: inversion and ambiguity removal. An example of this processing is provided for the single measurement point described by Table C.1.

Table C.1 Example Scatterometer Measurement Data

Parameter	Fore Antenna	Mid Antenna	Aft Antenna
Radar incidence angle	29.8°	21.7°	29.8°
Radar look angle	57.1°	102.9°	148.2°
Backscatter (σ°)	-9.08 dB	-2.16 dB	-8.71 dB
Noise figure (Kp)	6	5	6
Missing packet counter	0	0	0

INVERSION

All the data describing the example measurement (Table C.1) is passed to the subroutine WINRET. As the direction is stepped by 15° from 0° to 345°, the direction, σ° triplet, and incidence angle are passed to the function QSPEED to calculate the corresponding velocity, using coefficients from the CMOD-QSLUT look up table. The residual value, represented by equation (6), is then calculated in the function RESID. For the measurement values in Table C.1, the corresponding velocity and residual for each direction are given in Table C.2. The direction is then interpolated to the residual minima (shown highlighted in Table C.2) with a quadratic fit to the three points about each minimum. The subroutine FIT3 uses the direction values and residual values of the three points about each minimum to return coefficients to satisfy the equation

$$R = C_0 + C_1 * D + C_2 * D^2,$$

such that a new direction value is estimated at the minimum of the quadratic. The direction estimate is used in the function QSPEED to calculate a new corresponding velocity value for each minimum. The four direction and velocity solutions (Table C.3) are then passed

Table C.2 Velocity and Residual Values Corresponding to Given Direction Values (0°-345°)

Direction (°)	Velocity (m/s)	Residual
0	12.36	2.242
15	12.47	2.957
30	12.38	5.667
45	12.02	8.578
60	11.42	8.887
75	10.73	5.841
90	10.20	2.012
105	10.05	0.292
120	10.33	1.582
135	10.94	4.547
150	11.62	6.302
165	12.15	5.316
180	12.42	3.249
195	12.46	2.475
210	12.30	3.874
225	11.95	6.486
240	11.42	7.592
255	10.82	5.607
270	10.33	2.416
285	10.19	0.864
300	10.45	1.909
315	11.00	4.166
330	11.60	4.961
345	12.06	3.576

to the subroutine RKONRES, where the funcion RESID is again used tocalculate the residual values for each solution. The standard error of the measurement is calculated to be 0.7398 and is used to calculate the probability of each solution using the function PROB (Table C.3). Finally, the subroutine RANK ranks the solutions in decending order of relative probability. However, deficiencies in the CMOD4 model function cause the initial rank 1 probability to be an insufficient predictor of the most correct solution. Therefore, the rank 1 probability is re-calculated as a function of the number of valid solutions retrieved, the mid-beam incidence angle, and the retrieved (rank 1) wind speed in accordance with CMOD4. The final solution probabilities returned from WINRET are given in the last column of Table C.3.

Table C.3 Initial Wind Vector Solutions (from WINRET)

Direction (°)	Velocity (m/s)	Initial Probability	Re-calculated, Normailized Probability
106.1	10.05	0.916	0.490
286.5	10.20	0.770	0.224
2.27	12.39	0.511	0.149
192.8	12.47	0.471	0.137

AMBIGUITY REMOVAL

Background field comparison

The solutions and collocated NOGAPS model wind vector for the measurement point are then passed to the subroutine EXTFIT to compare and, if necessary, re-rank the scatterometer solutions. The corresponding NOGAPS wind vector for the example measurement point has a speed and direction of:

Wind speed = 12.06 m/s
Wind direction = 268.3°.

Using the standard deviation in the NOGAPS wind speed (3.206) and direction (20.0) and in the scatterometer speed (2.0) and direction (20.0) and the difference between the wind vectors from each source, the residuals of each scatterometer solution are re-calculated and re-ranked. The re-ranked solutions are shown in Table C.4.

Table C.4 Re-ranked Solutions (from EXTFIT)

Direction (°)	Velocity (m/s)	Re-calculated, Normailized Probability
286.5	10.20	0.973
192.8	12.47	0.0233
2.27	12.39	3.60E-3
106.1	10.05	1.82E-7

The SLICE Filter

The solutions for all continuous points are then passed to the SLICE filter for a "buddy check" to ensure a smooth final wind field. The filter sweeps along and across the swath in alternate directions, beginning with the first row at the outer edge of the swath. For each measurement point, the filter gathers the surrounding nodes within a 5 x 5 block. The block surrounding the example node is shown in Figure C.1. The rank 1 wind

direction values for each node in the block are shown in Table C.5. The direction values for each surrounding node are located in 45° sectors (from 0° to 360°) to ensure greater than 4 fall within a 135° range (i.e., 3 neighboring 45° sectors), see Table C.6. The local direction for the block is the average of all nodes falling with the three sectors and is calculated to be 288.4°. The difference between the local direction and the direction of the

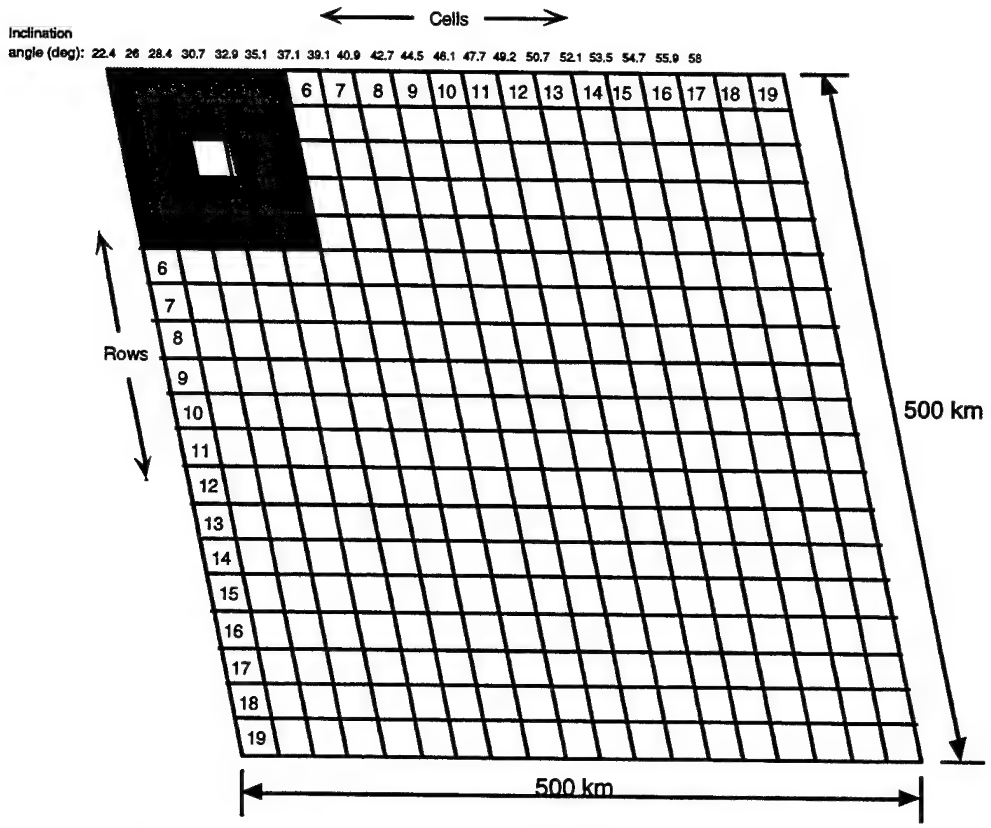

Cells
Inclination angle (deg): 22.4 26 28.4 30.7 32.9 35.1 37.1 39.1 40.9 42.7 44.5 46.1 47.7 49.2 50.7 52.1 53.5 54.7 55.9 58
6 7 8 9 10 11 12 13 14 15 16 17 18 19
Rows
6 7 8 9 10 11 12 13 14 15 16 17 18 19
500 km
500 km

Figure C.1 SLICE 5 x 5 Block for Example Node

center node is then used to recalculate the probability and then re-rank the solutions. The final ranking of the solutions of the example node is given in Table C.6.

Table C.5 5 x 5 Block Node Wind Directions (°)

Row/Cell	1	2	3	4	5
1	300.9	297.7	288.5	279.8	280.6
2	305.8	294.6	285.6	282.2	280.7
3	299.4	292.3	-	284.0	280.2
4	292.2	288.9	286.3	283.9	278.8
5	290.2	291.4	290.7	286.6	280.2

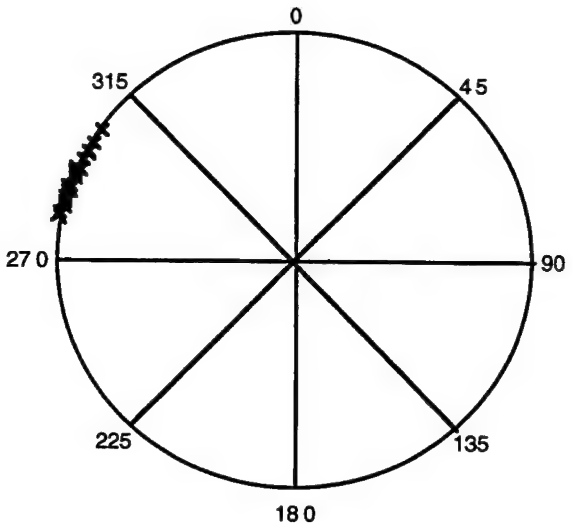


Figure C.2 5 x 5 Block Node Direction Distribution

Table C.6 Final Ranked Solutions (from SLICE)

Direction (°)	Velocity (m/s)	Re-calculated, Normailized Probability
286.5	10.20	0.965
192.8	12.47	0.032
2.27	12.39	0.003
106.1	10.05	0,000

APPENDIX D. FNMOC SCATTEROMETER PROCESSING CODE

```
      PROGRAM SCTR_ERS
C
C.............START   PROLOGUE..................................
C
C MODULE NAME:     ERS1SCAT
C
C DESCRIPTION:
C       This is the main program to process the decoded ERS-1/2 scatterometer data.
C       Processing occurs in two primary steps: quality control and wind retrieval.
C       Quality control consists of filtering data if:
C               a) it does not meet certain measurement standards
C               b) it is over land
C               c) it is over ice
C       Wind retrieval consists of inversion and ambiguity removal. Following wind
C       retrieval,data is written to the ISIS database.
C
C.............MAINTENANCE   SECTION..............................
C MODULES CALLED:
C
C NAME            DESCRIPTION
C ---------       ----------
C ROWCEL          Assign spacial row and cell numbers to each point
C DTGOPS          Retrieve the current watch dtg
C DTGMOD          Modify the dtg
C WINDOW          Checks data against valid time window
C QCHECK          Quality check data
C LANDMSK         Check data against land mask
C ICEMSK          Check data against ice mask
C WINDDATA        Retrieves wind speed and direction from ISIS
C DBSTOP          Closes the ISIS data base
C CONTIN          Checks data for continuity
C PROCESS         Formats and processes data
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME      TYPE        DESCRIPTION
C  --------  -------     -------------------
C BLK       INT         Number of total elements in a data block
C N         INT         Max expected # of records
C M         INT         Number of fields in each record
C P         INT         Number of records in a data block
C B         INT         Max expected # of blocks
C DTG       CHAR        Current watch dtg
C DTG3      CHAR        DTG with 3 hours added
C DTGNG3    CHAR        DTG with 3 hours subtracted
C DTG6      CHAR        DTG with 6 hours added
C DTGNG6    CHAR        DTG with 6 hours subtracted
C DTG9      CHAR        DTG with 9 hours added
C DTGNG9    CHAR        DTG with 9 hours subtracted
```

```
C DTG12      CHAR         DTG with 12 hours added
C RDTG       CHAR         Valid DTG
C IDTG3      INT          Integer DTG
C IDTGNG3    INT          Integer DTG
C IDTG9      INT          Integer DTG
C IDTGNG9    INT          Integer DTG
C IU         INT          Input unit number
C STAT       INT          Status returned from DTGMOD
C MOD3       INT          Three hours addition
C MODNG3     INT          Three hours subtraction
C MOD6       INT          Six hours addition
C MODNG6     INT          Six hours subtraction
C MOD9       INT          Nine hours addition
C MODNG9     INT          Nine hours subtraction
C MOD12      INT          Twelve hours addition
C RAWWND     INT          Raw wind
C DDATA      INT          Array of data
C QCDATA     INT          Array of data for quality control
C TDATA      INT          Array of data not w/in initial time window
C I, J, K    INT          Loop Variables
C ND         INT          Number of records in the array DDATA
C NFLD       INT          Counter for RAWWND
C NT         INT          Array counter
C NT2        INT          Array counter
C NQ         INT          Array counter
C NL         INT          Array counter
C NF         INT          Array counter
C NREC       INT          Array of record counts
C FTIME      INT          First record time: year, month, day, and hour
C FTIME2     INT          Record time of observations not w/in initial time window
C CONT       INT          Array of block continuity flags
C IBSAVE     INT          Number of blocks saved
C LAT        REAL         Array of valid latitudes
C LON        REAL         Array of valid longitudes
C WSPD       REAL         Array of wind speeds from ISIS
C WDIR       REAL         Array of wind directions from ISIS
C ERROR      INT          Error value
C TFLAG      INT          Data to be processed with new DTG
C MSG        INT          EXIT code variable for informative messages or 0 if
C                         completely successful run
C
C METHOD:
C       Read in raw wind data
C       Place in data array
C       Call ROWCEL to assign spacial row and cell numbers to each measurement point.
C       Call DTGOPS to get current watch DTG
C       Call DTGMOD to add and subtract 3, 6, 9, and 12 hours to watch time
C       Call WINDOW to check data against valid window
C       Call QCKECK to quality check the time valid data against user standards
C       Call LANDMSK to check qcdata against landmask
C       Call ICEMSK to retrieve  SSM/I ice data from ISIS
C       Call WINDDATA to retrieve wind data from ISIS
```

```
C       Call CONTIN to check filtered blocks for continuity
C       Call PROCESS for final formatting and processing
C     If TFLAG set then
C       mv FTIME2 to FTIME
C       mv NT2 to ND
C       mv TDATA to DATA
C       return to process remaining data
C       Call DBSTOP to clear the ISIS data base
C     Endif
CC
C  COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE....................................
C
      implicit none
C
      integer blck, n, m1, m2, p, b
      parameter(blck=15884,n=25000,m1=44,m2=33,p=361,b=85)
      character*16 dtg
      character*10 dtg3, dtgng3, dtg6, dtgng6, dtg9, dtgng9, dtg12, rdtg
      integer idtg3, idtgng3, idtg9, idtgng9
      integer iu
      integer mod3, modng3, mod6, modng6, mod9, modng9, mod12, stat
      integer rawwnd(blck), ddata(m1), qcdata(n,m2)
      integer i, j, k, nd, nfld, nt, nq, nl, nf, nrec(b)
      integer ftime
      integer tdata(n,m2), tflag, ftime2, nt2
      integer cont(b), ibsave
      real lat(n), lon(n)
      real wspd(n), wdir(n)
      integer error, msg
C
      msg=0
C
C  Read in data one block at a time (44 fields x 361 records = 15884). Write out each actual
C 44 field record to the array ddata and parse out data of interest while filling array qcdata.
C
      iu=10
C
      nd=0
C
 1000 continue
      read (iu,err=8888,end=1010,iostat=error) rawwnd
C
      nfld=0
C
      do 100 i=1,p
        nd=nd+1
        do 110 j=1,44
          nfld=nfld+1
          ddata(j)=rawwnd(nfld)
  110   continue
```

```
      qcdata(nd,1)=ddata(1)
      qcdata(nd,2)=ddata(5)
      do 120 j=3,29
        qcdata(nd,j)=ddata(j+15)
 120    continue
 100 continue
C
     go to 1000
1010 continue
C
C --------------------- Assign row and cell numbers --------------------
C  Assign spacial row and cell numbers to each measurement point.
C
     write(0,*) 'Calling rowcel.'
     call rowcel(qcdata,nd)
C
C ----------------- Determine appropriate synoptic DTG -----------------
C  Get current watch DTG.
C
     call dtgops(dtg,stat)
       dtg=dtg(1:10)
       write(0,*) 'Watch DTG is ',dtg
C
C  Pull start and end time of file.
C
     ftime=qcdata(1,4)*1000000 + qcdata(1,5)*10000 + qcdata(1,6)*100 + qcdata(1,7)
C
     write(0,*) 'File time is ',ftime
C
C  Use DTG of the data file (ftime) to determine most appropriate model run (synoptic)
C DTG,and adjust dtg if necessary.  This is for processing every 6 hours.
C
     mod3 = 3
     modng3 = -3
     mod6 = 6
     modng6 = -6
     mod9 = 9
     modng9 = -9
     mod12 = 12

     call dtgmod(dtg,mod3,dtg3,stat)       ! Create dtg+3
     read(dtg3,'(i10)') idtg3              ! Change from character to integer
     call dtgmod(dtg,modng3,dtgng3,stat)   ! Create dtg-3
     read(dtgng3,'(i10)') idtgng3          ! Change from character to integer
     call dtgmod(dtg,mod6,dtg6,stat)       ! Create dtg+6
     call dtgmod(dtg,modng6,dtgng6,stat)   ! Create dtg-6
     call dtgmod(dtg,mod9,dtg9,stat)       ! Create dtg+9
     read(dtg9,'(i10)') idtg9              ! Change from character to integer
     call dtgmod(dtg,modng9,dtgng9,stat)   ! Create dtg-9
     read(dtgng9,'(i10)') idtgng9          ! Change from character to integer
     call dtgmod(dtg,mod12,dtg12,stat)     ! Create dtg+12
C
```

```
 6666 continue
C
     if (ftime.ge.idtg9) then            ! Compare file dtg with watch dtg+9
       rdtg=dtg12                        ! Add 12 to watch dtg
     else if (ftime.ge.idtg3) then       ! Compare file dtg with watch dtg+3
       rdtg=dtg6                         ! Add 6 to watch dtg
     else if (ftime.lt.idtgng9) then     ! Compare file dtg with watch dtg-9
       write(0,*) 'MESSAGE: File time older than: ',idtgng9
       call EXIT(22)                     ! File too old
     else if (ftime.lt.idtgng3) then     ! Compare file dtg with watch dtg-3
       rdtg=dtgng6                       ! Subtract 6 from watch dtg
     else
       rdtg=dtg
     endif
C
C ------------ Check against valid time window and time sort blocks ------------
C  Check data against valid window +-3 hours from the appropriate synoptic dtg.
C  Also time sort blocks and check for duplicity of blocks.
C
      write(0,*) 'Calling window.'
     call window(qcdata,rdtg,nd,nt,tdata,tflag,ftime2,nt2)
C
C ----------------------- Perform all quality control checks -----------------------
C  Check data against user selected standards for noise figure and missing packets and
C check if any confidence flags are turned on.  Return only data completely within
C standards. Also check for and filter out any records with any field assigned a decoder
C 'missing' value.
C
      write(0,*) 'Calling qcheck.'
     call qcheck(qcdata,nt,lat,lon,nq)
C
C  Check data against land mask and only return data not over land.
C
      write(0,*) 'Calling landmsk.'
     call landmsk(qcdata,lat,lon,nq,nl,msg)
C
C  Check data against current SSM/I ice analysis and only return data not over ice.
C
C  * Note: ice analysis is only written to ISIS every 12 hours; therefore,
C      the unmodified dtg is used.
C
      write(0,*) 'Calling icemsk.'
     call icemsk(qcdata,lat,lon,nl,dtg,nf)
C
C  At this point, all remaining data have passed all quality checks and will be passed to the
C wind retrieval phase.
C
C  If no data remains, end process.
     if (nf.eq.0) then
 7770    continue
       write(0,*) 'MESSAGE: No data remains from qc to process.'
       msg=2
```

```
        go to 7777
      endif
C
C ---------------------- Colocate model output -----------------------
C  Colocate with current NOGAPS wind speed and direction.
C  * Note: wind data is written to ISIS every 6 hours, so the modified
C       dtg (ie. rdtg) can be used to pull the data.
C
       write(0,*) 'Calling winddata.'
       call winddata(lat,lon,nf,rdtg,wspd,wdir)
C
C -------------------- Perform continuity check ----------------------
C  Check filtered blocks for continuity and assign flag to each block for processing.
C
       write(0,*) 'Calling contin.'
      call contin(qcdata,nf,ibsave,nrec,cont)
C -----------------------------------------------------------------
C                       Begin Processing
C -----------------------------------------------------------------
C  Final formatting and processing takes place in the subroutine "process".
C
       write(0,*) 'Calling process.'
      call process(qcdata,wspd,wdir,ibsave,nrec,nf,cont,msg)
C
C  * Note:  Data is written to ISIS in "process".
C -----------------------------------------------------------------
 7777 continue
C
C  If TFLAG is set, return to process remaining data with a new time window.
C
      if (tflag.eq.1) then
        ftime=ftime2
        nd=nt2
        do 200 i=1,nd
          do 210 j=1,31
            qcdata(i,j)=tdata(i,j)
 210        continue
 200      continue
        go to 6666
      endif
C -----------------------------------------------------------------
C  Call ISIS utility to close access to ISIS database.
C
      call dbstop()
C -----------------------------------------------------------------
      go to 9999
 8888 write(0,*) 'FATAL ERROR: Error reading scatterometer file:
     + IOSTAT = ',error
      call EXIT(20)
 9999 continue
      call EXIT(msg)
      END
      SUBROUTINE  ROWCEL(DATA,ND)
       use CONSTANTS
C
C.............START   PROLOGUE..................................
C
C MODULE NAME:       ROWCELL
C
C DESCRIPTION:
C       This subroutine assigns row and cell numbers to each measurement point.  Both
C       row # and cell # count from 1 to 19 as the antenna incidence angle moves through
C       19 values crossing the width of the swath. 19 rows of sweeps across the swath
C       cover a 500km x 500km frame, called a block.  The appropriate row and cell
C       numbers are then added to each record in the data array.
C
C USAGE:    CALL ROWCEL(DATA, ND)
C
C PARAMETERS:
C   NAME             TYPE          USAGE          DESCRIPTION
C -------------      ----------    ----------     --------------------
C  DATA              INT           IN/OUT         Field data array
C  ND                INT           IN/OUT         Number of data points
C
C.............MAINTENANCE   SECTION..............................
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME       TYPE          DESCRIPTION
C --------   --------      -----------
C N          INT           Max expected # of records
C M          INT           Number of fields in each record
C ROW        INT           Array of row numbers
C CELL       INT           Array of cell Numbers
C IROW       INT           Row number
C ICEL       INT           Cell number
C INCANG     REAL          Inclination angle
C ANGMIN     REAL          Minimum angle for a cell
C ANGMAX     REAL          Maximum angle for a cell
C I, R, C, J   INT         Loop variables
C DATA_ERR   INT           EXIT code from CONSTANTS file
C
C METHOD:
C       Assign row and cell numbers a block at a time.
C       The first inclination angle of each record is compared to the min and max values in
C       the arrays ANGMIN and ANGMAX, representing location across the swath. This
C       position determines the cell number (1-19). Write row and cell numbers to the data
C       array as fields 30 and 31.
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE....................................
C
      implicit none
```

```
C
    integer n, m
    parameter(n=25000,m=33)
    integer data(n,m), nd, row(n), cell(n)
    integer irow, icell
    real incang, angmin(19), angmax(19)
    integer i, r, c, j
C
     data  angmin/22.4,26.0,28.4,30.7,32.9,35.05,37.1,39.05,40.9,
    +          42.7,44.45,46.1,47.7,49.2,50.65,52.1,53.45,54.7, 55.9/,
    +     angmax/26.0,28.4,30.7,32.9,35.05,37.1,39.05,40.9,42.7,
    +          44.45,46.1,47.7,49.2,50.65,52.1,53.45,54.7,55.9, 58.0/
C
    i=0
C
C  Assign row and cell numbers a block at a time.  First inclination angle of each record is
C compared to min and max values listed above, representing location across the swath.
C This position determines the cell number (1-19).
C
 1000 continue
    do 100 r=1,19
      do 110 c=1,19
        i=i+1
        if (i.gt.nd) go to 2000          ! Leaves loop at end of records
        incang=data(i,12)/10.                ! 1st inclination angle
        do 120 j=1,19
          if (incang.ge.angmin(j) .and. incang.lt.angmax(j)) then
            icell=j                      ! Assigns cell number
            if (i.eq.1) then               ! Assigns row number
              irow=1
            elseif (icell.le.cell(i-1)) then
              irow=row(i-1) + 1
              if (irow.gt.19) irow=1         ! New block
            else
              irow=row(i-1)
            endif
            row(i)=irow
            cell(i)=icell
            go to 1000
          endif
 120      continue
C
C  Should never reach this section: reading for inclination angle out of expected range
C
    write(0,*) 'FATAL ERROR: Inclination out of range: angle = ',incang
    call EXIT(DATA_ERR)
C
 110    continue
 100 continue
    if (i.lt.nd) go to 1000
2000 continue
C
```

```
C  Write row and cell numbers to the data array as fields 30 and 31.
C
      do 200 i=1,nd
        data(i,30)=row(i)
        data(i,31)=cell(i)
  200 continue
C
      END
```

```
SUBROUTINE WINDOW(DATA,RDTG,ND,NT,TDATA,TFLAG,FTIME2,NT2)
C
C..............START   PROLOGUE...................................
C
C MODULE NAME:      WINDOW
C
C DESCRIPTION:
C      Checks time of observations and compares to the valid window(+/- 3 hours of
C      model run).  If time is valid, the record is saved in the data array. Also, the
C      data is prepared for sorting.  Then time sorts blocks and checks for duplicity.
C
C USAGE:  CALL WINDOW(DATA,RDTG,ND,NT,TDATA,TFLAG,FTIME2,NT2)
C
C PARAMETERS:
C   NAME           TYPE        USAGE      DESCRIPTION
C -------------    ----------  -------    --------------------
C DATA             INT         IN         Data observations
C RDTG             CHAR        IN         Time of observations
C ND               INT         IN         Number of data points
C NT               INT         OUT        Number of sorted points
C TDATA            INT         OUT        Data observations saved for later
                                          processing
C TFLAG            INT         OUT        Flag to indicate observations outside
C                                         time window
C FTIME2           INT         OUT        Time of observations in TDATA
C NT2              INT         OUT        Number of points in TDATA
C
C..............MAINTENANCE   SECTION...............................
C
C MODULES CALLED:
C
C NAME      DESCRIPTION
C ----      -----------
C DTGMOD    Modifies the DTG
C TMSORT    Sorts and checks blocks for duplicity
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME      TYPE        DESCRIPTION
C ---------  -------    -----------
C N          INT        Max expected # of records
C M          INT        Number of fields in each record
C P          INT        Number of records in data block
C B          INT        Max expected # of blocks
C DATA3D     INT        Intermediate sorted data array
C RDTG1      CHAR       Minus 3 hours date time group
C RDTG2      CHAR       Plus 3 hours date time group
C MOD3       INT        Plus 3 hours variable
C MODNG3     INT        Minus 3 hours variable
C IDTG1      INT        Integer RDTG1
C IDTG2      INT        Integer RDTG2
C STAT       INT        Status return from DTGMOD
```

```
C BLCK      INT           Array of block numbers
C BTIME     INT           Time of block measurement
C  I,J,K,R,L INT          Loop counters
C IB        INT           Block counter
C DDTG      INT           DTG of each record
C DTIME     INT           Time of each record
C INTM      INT           Time of each record, used for comparison
C IBB       INT           Block counter
C NR2       INT           Number of records in each block
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE...................................
C
      implicit none
C
       integer n, m, p, b
       parameter(n=25000, m=33, p=361, b=70)
       integer data(n,m), tdata(n,m), data3d(p,m,b), nd
      character*10 rdtg, rdtg1, rdtg2
      integer mod3, modng3, idtg1, idtg2, stat
      integer blck(b), btime(b), nr(b)
       integer i, j, k, l, r, ib
      integer ddtg, dtime, intm, ftime2
       integer ibb, nr2(b), nt, tflag, nt2
C
      mod3=3
      modng3=-3
      tflag=0
C
C --------------------- Check against valid window -----------------
C
C Make valid window for data +/- 3 hours from appropriate synoptic DTG.
C
      call dtgmod(rdtg,modng3,rdtg1,stat)
      read(rdtg1,'(i10)') idtg1         ! Convert character to integer
      call dtgmod(rdtg,mod3,rdtg2,stat)
      read(rdtg2,'(i10)') idtg2         ! Convert character to integer
       write(0,*) 'The window for valid data is between ',rdtg1,' and ',
     + rdtg2
C
C Read time of all observations and compare to valid window. If valid, save record in
C array.  Also write out block number, block time and records/block for future sorting.
C
C If data is outside +- 3 hour window, save record time and turn tflag on.  When
C processing is complete, if tflag is turned on, return to recalculate new appropriate model
C run DTG for unprocessed data and begin process again for rest of the data.
C
      j=0
      l=0
      ib=0
      intm=-999
```

```
    do 100 i=1,nd
      ddtg=data(i,4)*1000000 + data(i,5)*10000 + data(i,6)*100 + data(i,7)      ! Data dtg
 dtime=(data(i,4)*data(i,5)*data(i,6)*24*3600) + data(i,7)*3600+ data(i,8)*60+
   +      data(i,9)/1000                                                    ! Data time (sec)
      if (ddtg.ge.idtg1.and.ddtg.lt.idtg2) then                             ! Within window
        j=j+1
        do 110 k=1,31
          data(j,k)=data(i,k)
 110      continue
        if (intm.ne.dtime) then                      ! New block
          if (ib.gt.0) nr(ib)=r            ! Saves # of records/block
          ib=ib+1                         ! Counts # of blocks
          blck(ib)=ib                     ! Saves block #
          btime(ib)=dtime                    ! Saves time of block
          r=0                            ! Resets record counter
          intm=dtime
        endif
        r=r+1                             ! Counts # of records/block
      else
        l=l+1
        if (l.eq.1) ftime2=ddtg        ! Save record time for future processing
        tflag=1                    ! Indicates data to process w/ different DTG
        do 120 k=1,31
          tdata(l,k)=data(i,k)
 120      continue
      endif
 100 continue
     nr(ib)=r                      ! Saves # of records for last block
     nt2=l
     if (j.eq.0) then
       return 7770
     endif
C
C -------------------- Time sort and block checks ----------------
C  Call subroutine to sort and check blocks for duplicity.
C
     call tmsort(data,blck,btime,nr,ib,ibb,nr2,data3d)
C
C  Convert 3D array data3d back to the 2D array data for future filtering, writing out
C (sequentially) only non-duplicate blocks(index ne 0).
C
     r=0
     do 200 i=1,ibb
       do 210 j=1,nr2(i)
         r=r+1
         do 220 k=1,32
           data(r,k)=data3d(j,k,i)
 220       continue
 210     continue
 200 continue
     nt=r
     END
```

```
      SUBROUTINE TMSORT(DATA,BLCK,BTIME,NR,IB,IBB,NR3,DATA3D)
C
C..............START   PROLOGUE...................................
C
C MODULE NAME:       TMSORT
C
C DESCRIPTION:
C       The subroutine sorts the blocks of data by time and checks for duplicity.
C
C USAGE:    CALL TMSORT(DATA, BLCK, BTIME, NR, IB, IBB, NR3, DATA3D)
C
C PARAMETERS:
C   NAME     TYPE          USAGE        DESCRIPTION
C ----------- -------       ----------   -------------------
C DATA       INT           IN           Array of data
C BLK        INT           IN           Number of blocks per data seq.
C BTIME      INT           IN           Time of block
C NR         INT           IN           Array of number of records
C IB         INT           IN           Number of blocks
C IBB        INT           OUT          New number of blocks
C NR3        INT           OUT          New number of records
C DATA3D     INT           OUT          3 dimensional data array
C
C
C..............MAINTENANCE   SECTION..............................
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME       TYPE          DESCRIPTION
C --------   --------      -------------------
C N          INT           Max expected # of records
C M          INT           Number of fields in each record
C P          INT           Number of records in data block
C B          INT           Max expected # of blocks
C INDEX      INT           Time index for blocks of data
C INDEX2     INT           Time index for blocks of data
C NR2        INT           Indexed number of records
C SS1        INT           Hour, minutes, and seconds of data in seconds
C SS2        INT           Hour, minutes, and seconds of data in seconds
C DY1        INT           Day of data
C DY2        INT           Day of data
C DIF        INT           Time difference
C I, J, K, R INT           Loop variables
C IND        INT           Index number when time difference is > 30 seconds
C
C METHOD:
C       Sort blocks by time
C       Convert 2D array to 3D array, dimensioned by record number and block number.
C       Check if blocks start within 30 seconds of each other.  If so, assume duplicate.
C       Assign duplicate block a bogus block number, correct remaining block numbers
C       and filter out duplicate block.
C
```

```
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE...................................
C
   implicit none
C
    integer n, m, p, b
    parameter(n=25000,m=33,p=361,b=70)
    integer data(n,m), data3d(p,m,b), blck(b)
    integer btime(b), nr(b), index(b), index2(b), nr2(b), nr3(b)
    integer ss1, ss2, dy1, dy2, dif
    integer i, j, k, ib, ibb, r, ind
C
C Sort blocks by time.
C
    do 100 i=1,ib
      index(i)=blck(i)              ! Initially assign index=blck
      if (blck(i).eq.1) then         ! Begin sequential check
        index(i)=1
        go to 130
      else
        do 110 j=1,blck(i)-1
          if (btime(j).gt.btime(blck(i)) .and. index(j).lt.index(i)) then      ! Out of order
            index(i)=index(j)              ! Reassigns index #
          endif
 110      continue
        do 120 k=1,i-1                  ! Updates other index
          if (index(k).ge.index(i)) then      !  #'s as appropriate
            index(k)=index(k)+1
          endif
 120      continue
        go to 130
      endif
 130  continue
      nr2(i)=nr(index(i))        ! Re-associates # of records w/ index
 100 continue
C
C Future block reference will use 'index'.
C
C Convert 2D array to 3D array, dimensioned by record number and block number.
C
   r=1
   do 200 i=1,ib
     do 210 j=1,nr2(i)
       do 220 k=1,31
         data3d(j,k,index(i))=data(r,k)
 220     continue
       r=r+1
 210   continue
 200 continue
C
C Check if blocks start within 30 seconds of each other.  If so, assume duplicate.  Assign
```

```
C  duplicate block a bogus block number, correct remaining block numbers and
C  filter out duplicate block. The 'do' loop will pull out blocks sequentially by index
C  number because the blocks were dimensioned using index values.
C
     ind=1
     do 300 i=1,ib
       if (i.eq.1) then
          ss1=data3d(1,7,1)*3600 + data3d(1,8,1)*60 + data3d(1,9,1)/1000
          dy1=data3d(1,6,1)
          index2(i)=1
          nr3(i)=nr2(index(i))
       else
          ss2=data3d(1,7,i)*3600 + data3d(1,8,i)*60 +
     +         data3d(1,9,i)/1000          ! Time (sec) of current block
          dy2=data3d(1,6,i)
          if (dy1.lt.dy2) ss2=ss2 + 3600*24
          dif=ss2-ss1
          if (dif.le.30) then            ! Checks for duplicity
            index2(i)=0                  ! Assigns bogus index number
          else
            ind=ind+1
            if (ss2.gt.3600*24) then   ! Saves block time for next comparison
              ss1=ss2-3600*24          !  (checking if ss2 was changed
            else                     !   because of day change)
              ss1=ss2
            endif
            dy1=dy2
            index2(i)=ind              ! Reassigns index number to index2
            nr3(i)=nr2(index(i))        ! Re-associates # of records
          endif                      !   w/ index
       endif
 300 continue
     ibb=ind                        ! Resets # of blocks
C
C  Write index (block) number to data array as field 32 and re-dimensionalize
C  according to index2.  Again the blocks are read sequentially by index
C  number.
C
     do 400 i=1,ib
          do 410 j=1,nr3(i)
            do 420 k=1,31
              data3d(j,k,index2(i))=data3d(j,k,i)
 420          continue
            data3d(j,32,index2(i))=index2(i)
 410      continue
 400 continue
     END
```

```
      SUBROUTINE  QCHECK(DATA,ND,LAT,LON,NMS)
C
C.............START   PROLOGUE...................................
C
C  MODULE NAME:      QCHECK
C
C  DESCRIPTION:
C        This subroutine quality checks the data against user selected standards.  If any part
C        of the data falls outside of standards, the entire record is filtered out. Missing values
C        assigned by the decoder are also checked and associated records are filtered out.
C
C  USAGE:    CALL QCHECK(DATA, ND, LAT, LON, NMS)
C
C  PARAMETERS:
C    NAME     TYPE          USAGE        DESCRIPTION
C   ---------  -------       ----------    ---------------------
C  DATA       INT           IN           Array of data
C  ND         INT           IN           Number of data points
C  LAT        REAL          OUT          Array of acceptable lat points
C  LON        REAL          OUT          Array of acceptable lon points
C  NMS        INT           OUT          Number of acceptable points
C
C.............MAINTENANCE   SECTION..............................
C
C  MODULES CALLED:
C
C  NAME      DESCRIPTION
C  --------  ------------------
C  CONFID    Check confidence flags and returns acceptable data
C
C  LOCAL VARIABLES AND STRUCTURES:
C
C  NAME      TYPE           DESCRIPTION
C  --------  -------        ------------------
C  N         INT            Max expected # of records
C  M         INT            Number of fields in each record
C  DMDATA    INT            Data with decoder missing values filtered out
C  CFDATA    INT            Data with that has been confidence checked
C  KPDATA    INT            Data that has had the noise figure filtered out
C  MS1       INT            Missing packet numbers for antenna 1
C  MS2       INT            Missing packet numbers for antenna 2
C  MS3       INT            Missing packet numbers for antenna 3
C  MSMAX     INT            Maximum allowable missing packets
C  MSTOT     INT            Total number of missing packets
C  KP1       REAL           Noise figure for antenna 1
C  KP2       REAL           Noise figure for antenna 2
C  KP3       REAL           Noise figure for antenna 3
C  I, J, K, F INT           Loop variables
C  NDM       INT            Total number of DMDATA values
C  NCF       INT            Total number of CFDATA values
C  NKP       INT            Total number of KPDATA values
C
```

```
C METHOD:
C       Filter out records with the assigned decoder missing value Perform CONFID to
C       check confidence flags and return acceptable data Filter out data with noise figure
C        (Kp) greater than 10%. Filter out data with total missing packets greater than 15.
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C..............END   PROLOGUE......................................
C
      implicit none
C
      integer n, m
      parameter(n=25000,m=33)
      integer data(n,m), dmdata(n,m), cfdata(n,m), kpdata(n,m), nd
      integer ms1, ms2, ms3, msmax, mstot
      real kp1, kp2, kp3
      integer i, f, j, k, ndm, ncf, nkp, nms
      real lat(n), lon(n)
C
C Filter out records with fields assigned decoder 'missing' value of -32768.
C
      j=0
      do 100 i=1,nd
        do 110 f=1,29
          if (data(i,f).eq.-32768) go to 100
 110    continue
        j=j+1
        do 120 k=1,32                    ! Write out acceptable records
          dmdata(j,k)=data(i,k)
 120    continue
 100 continue
      ndm=j
C
C Check confidence flags of data and only return acceptable data
C
      call confid(dmdata,ndm,cfdata,ncf)
C
C Filter out data with noise figure (Kp) greater than 10%.
C
      j=0
      do 200 i=1,ncf
        kp1=cfdata(i,15)/10.
        kp2=cfdata(i,20)/10.
        kp3=cfdata(i,25)/10.
          if (kp1.ge.0.and.kp1.le.10 .and. kp2.ge.0.and.kp2.le.10 .and.
+        kp3.ge.0.and.kp3.le.10) then
          j=j+1
          do 210 k=1,32                  ! Write out acceptable records
            kpdata(j,k)=cfdata(i,k)
 210      continue
        endif
 200 continue
```

```
      nkp=j
C
C  Filter out data with total missing packets greater than 15.
C
      msmax=15
      j=0
      do 300 i=1,nkp
        ms1=kpdata(i,16)
        ms2=kpdata(i,21)
        ms3=kpdata(i,26)
        mstot=abs(ms1)+abs(ms2)+abs(ms3)
        if (mstot.le.msmax) then
          j=j+1
          lat(j)=float(kpdata(i,10))/100.0  ! Write out latitudes &
          lon(j)=float(kpdata(i,11))/100.0  !  longitudes of acceptable records
          do 310 k=1,32
            data(j,k)=kpdata(i,k)           ! Write out acceptable records
 310        continue
        endif
 300  continue
      nms=j
      END
      SUBROUTINE  CONFID(DATA,ND,CFDATA,NCF)
C
C MODULE NAME:       CONFID
C
C DESCRIPTION:
C       This subroutine checks the bits of the confidence flags at each measurement point
C       and returns an err=1 if any are turned on.
C
C USAGE:  CALL CONFID(DATA, ND, CFDATA, NCF)
C
C PARAMETERS:
C   NAME     TYPE        USAGE       DESCRIPTION
C   --------  --------    ----------  -------------------
C DATA       INT         IN          Quality check data array
C ND         INT         IN          Number of data elements
C CFDATA     INT         OUT         Confidence data array
C NCF        INT         OUT         Number of confidence elements
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME        TYPE   DESCRIPTION
C --------    ------- -------------------
C fb          INT    Fore beam sigma presence
C mb          INT    Mid beam sigma presence
C ab          INT    Aft beam sigma presence
C fba         INT    Fore beam arcing
C mba         INT    Mid beam arcing
C aba         INT    Aft beam arcing
C kp          INT    Noise figure (Kp) > 20%
C ls          INT    Land contamination
C fc          INT    Checksum error detection
C mis         INT    Flags present
C conf        INT    Confidence array
C err         INT    Error flag array
C i, j, k     INT    Loop variables
C
C METHOD:
C       Move the confidence elements out of the data array into a confidence array.
C       Loop on number of confidence elements
C         Test the bits of each elements in the confidence array for flag bits being set.
C         If any bits have been set for a particular element then set the error flag to 1
C         Else
C           set the error flag to 0
C         Endif
C       Endloop
C       Write out a new data array where none of the confidence flags have been set.
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE....................................
C
  implicit none
```

```
C
      integer n, m
      parameter(n=25000,m=33)
      integer data(n,m), cfdata(n,m), nd
      integer err(n), i, j, k, ncf
      integer fb, mb, ab, fba, mba, aba, kp, ls, fc, mis
C
C  The following describes the bits in the confidence flag:
C   fb, mb, ab => fore, mid, and aft beam sigma presence
C   fba, mba, aba => fore, mid, and aft beam arcing
C   kp => Kp > 20%
C   ls => land contamination
C   fc => checksum error detection
C   mis => flags present
C
      do 100 i=1,n                    ! Decode flag bits for check
        fb = mod(data(i,29)     ,2)
        mb = mod(data(i,29)/2   ,2)
        ab = mod(data(i,29)/4   ,2)
        fba= mod(data(i,29)/8   ,2)
        mba= mod(data(i,29)/16  ,2)
        aba= mod(data(i,29)/32  ,2)
        kp = mod(data(i,29)/64  ,2)
        ls = mod(data(i,29)/128 ,2)
        fc = mod(data(i,29)/2048,2)
        mis= mod(data(i,29)/4096,2)
C
        if (mis.gt.0 .or. fb .gt.0 .or.
     +      mb .gt.0 .or. ab .gt.0 .or.
     +      fba.gt.0 .or. mba.gt.0 .or.
     +      aba.gt.0 .or. kp .gt.0 .or.
     +      ls .gt.0 .or. fc .gt.0) then
          err(i)=1
        else
          err(i)=0
        endif
 100  continue
C
C  Filter out data with any confidence flag bits turned on.
C
      j=0
      do 200 i=1,nd
        if (err(i).eq.0) then
          j=j+1
          do 210 k=1,32
            cfdata(j,k)=data(i,k)
 210      continue
        endif
 200  continue
      ncf=j
      END
      SUBROUTINE  LANDMSK(DATA,LAT,LON,ND,NL,MSG)
     use CONSTANTS
C
C CONFIGURATION IDENTIFICATION:      ICEMSK
C
C MODULE NAME:
C
C DESCRIPTION:
C        This subroutine takes the decoded data array and compares it to a land-sea mask.
C         The data records over land are then filtered from the array.
C
C USAGE:      CALL LANDMSK(DATA,LAT,LON,ND,NL,MSG)
C
C PARAMETERS:
C   NAME    TYPE          USAGE        DESCRIPTION
C   ---------  -------         ----------        ---------------------
C   DATA    INT            IN              Array of scatter data
C   LAT      FLOAT        IN              Array of latitude points
C   LON      FLOAT        IN              Array of longitude points
C  ND        INT            IN              Number of DATA elements
C   NL        INT            OUT           Number of elements over water
C   MSG     INT            OUT           Message EXIT code
C
C MODULES CALLED:
C
C NAME      DESCRIPTION
C ---------      ------------------
C LLSEA     Checks scatterometer measurement points against land database
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME                  TYPE  DESCRIPTION
C  --------                 -------  ------------------
C N                        INT    Max expected # of records
C M                        INT    Number of fields in each record
C  I, J, K                INT    Loop parameters
C  TLEN                 INT    Length of array TAB, dependent on resolution
C  STATUS             INT    Status of return from LLSEA
C  TAB                   INT    Array used for the land/sea table, read by LLSEA
C  RES                  REAL  Resolution in meters at the equator
C  LS                     LOG   Array of logical output for type found at corresponding
C                                   points in arrays LAT and LON
C  SEQTYPE          CHAR Table type
C  PATHNM            CHAR Source of table
C LAND_ERR         INT    EXIT code from CONSTANTS file
C
C METHOD:
C             Initialize variables
C                  Call LLSEA to read logical variables for each point (lat/lon)
C                  Check each point for land contamination
C                  If LS array is false (no land) then
C              Increment the j counter
```

```
C              Save lat and lon of the point
C              Save the specific data element in the data array
C            Endif
C            Move j to nl
C
C  COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE....................................
C
     implicit none
C
     integer n, m
     parameter(n=25000,m=33)
     integer data(n,m), nd
     integer i, j, k, tlen, status, nl, msg
     parameter(tlen=500000)
     integer tab(tlen)
     real lat(n), lon(n), res
     parameter(res=1000.)
     logical ls(n)
     character*24 seqtype
     parameter(seqtype='lnd')
     character*200 pathnm
     parameter(pathnm='isis')
C
C  Read logical variable for land at each lat/lon.
C
     call llsea(seqtype,res,tab,tlen,lat,lon,nd,pathnm,ls,status)
C
C  If land variable is false the associated point is over water and is writen out to array data.
C
     if (status.eq.0) then
       j=0
       do 100 i=1,nd
         if (.not.ls(i)) then
           j=j+1
           lat(j)=lat(i)
           lon(j)=lon(i)
           do 110 k=1,32
             data(j,k)=data(i,k)
110        continue
         endif
100    continue
       nl=j
     else
       write(0,*) 'MESSAGE: Error in LLSEA: status= ',status
       nl=nd
       msg=LAND_ERR
     endif
     END
```

```
      SUBROUTINE  ICEMSK(DATA,LAT,LON,ND,DTG,NI)
      use CONSTANTS
C
C..............START   PROLOGUE...................................
C
C  MODULE NAME:      ICEMSK
C
C  DESCRIPTION:
C       This subroutine reads global ice coverage in percentageshen interpolates the values
C       to the desired points.
C
C       Ice analysis data is only written to ISIS every 12 hours. It is generally written to
C       ISIS at 0Z,12Z + 2hrs. The data is only searched for at a forcast period of 0.
C
C  USAGE:    CALL ICEMSK(DATA,LAT,LON,ND,DTG,NI)
C
C  PARAMETERS:
C    NAME     TYPE          USAGE        DESCRIPTION
C    --------  --------      ----------   -------------------
C    DATA     INT           IN           Array of scatter data
C    LAT      FLOAT         IN           Array of latitude points
C    LON      FLOAT         IN           Array of longitude points
C    ND       INT           IN           Number of DATA elements
C    DTG      CHAR          IN           Date-Time-Group
C    NT       INT           OUT          Number of ice elements
C
C  MODULES CALLED:
C
C  NAME      DESCRIPTION
C  ---------  -----------
C  GGRD      Place grid parameters into a common field
C  VLLXY     Convert latitude and longitudes to x and y
C  PFLDID    Place data parameters into common fields
C  GETFLD    Request ISIS fields
C  DTGMOD    Modify a date-time-group
C  FINTRP    Interpolate isis data to wind data measurement points
C
C  LOCAL VARIABLES AND STRUCTURES:
C
C  NAME      TYPE          DESCRIPTION
C  ---------  -------       -------------------
C  N         INT           Max expected # of records
C  M         INT           Number of fields in each record
C  GEOM1     INT           Pointer
C  GEOM2     INT           Pointer
C  VSTAT1    INT           Status of return
C  VSTAT2    INT           Status of return
C  UNITLL    CHAR          Units of lat and lon for use in VLLXY
C  IFLD1     INT           Pointer for PFLDID
C  IFLD2     INT           Pointer for PFLDID
C  ISTGR     INT           Set to 0 if field is scaler
C  IWRP      INT           Set to 1 if field is periodic, 0 for regional map
```

```
C  STAT       INT       Status of return
C  STAT1      INT       Status of return for PFLDID
C  STAT2      INT       Status of return for PFLDID
C  LVLCNT     INT       Level count
C  ILEN1      INT       Number of points to be interpolated
C  ILEN2      INT       Number of points to be interpolated
C  LVL1       FLOAT     First level of ISIS field parameter
C  LVL2       FLOAT     Second level of ISIS field parameter
C  ZLVL       FLOAT     Array containing the depth/height values for each lvlcnt
C  FCSTPER    FLOAT     Forcast period
C  FLD1       FLOAT     Array filled with grid data
C  FLD2       FLOAT     Array filled with grid data
C  MDLTYPE    CHAR      Model type
C  GEOMNM1    CHAR      GEOM name
C  GEOMNM2    CHAR      GEOM name
C  PARMNM     CHAR      Parameter name
C  UNITS      CHAR      Data units requested
C  DSETNM     CHAR      Data set requested
C  LVLTYPE    CHAR      Level type as specified in geomnm
C  PATHNM     CHAR      Path name
C  SUFFIX     CHAR
C  SECLEVL    CHAR      Security level
C  DTG        CHAR      Date time group
C  REMARK     CHAR
C  TITLE      CHAR
C  X          FLOAT     Array of X values converted from latitudes
C  Y          FLOAT     Array of Y values converted from longitudes
C  SHEM       FLOAT     Southern hemisphere interpolated ice
C  NHEM       FLOAT     Northern hemisphere interpolated ice
C  ICE        FLOAT     Array of ice data
C  MWRK       INT       Actual first dimension of ice grid (normally = min)
C  MIN        INT       First dimension of ice grid
C  NIN1       INT       Second dimension of ice grid
C  NIN2       INT       Second dimension of ice grid
C  IFLAGI     INT       Flag specifying if input fields may contain undefined values
C                         0 = no,  1 = yes
C  FVALI      FLOAT     Value expected in input to specify undefined points
C  FVALO      FLOAT     Value in output to specify undefined points
C  FILVAL     FLOAT     Value in output to specify field off the input grid
C  I, J, K    INT       Loop parameters
C  MOD        INT       DTG modifier
C  SCOUNT     INT       Count of how far back in time (12 hr increments) your
C                       looking for the SHEM
C  NCOUNT     INT       Count of how far back in time (12 hr increments) your
C                       looking for the NHEM
C  DTGN12     CHAR      Modified date
C  SDTG       CHAR      Southern hemisphere modified date
C  NDTG       CHAR      Northern hemisphere modified date
C  ICE_ERR    INT       EXIT code from CONSTANTS file
C
C  METHOD:
C             Initialize variables
```

```
C              Work  on requesting Southern Hemisphere data
C              Call GGRD to place grid parameters into a common field
C              Call VLLXY to convert latitude and longitude into x and y
C              Call PFLDID to place ISIS calling parameters into a
C                     common field
C              Call GETFLD to retrieve ice data from ISIS
C              If GETFLD was unsuccessful then
C                Call DTGMOD to modify date to 12 hours back
C                Increment the SCOUNT
C                IF SCOUNT equals 3 then
C                  Print No ice available message
C                  Go to Northern Hemisphere requests
C                ENDIF
C                GO TO GGRD
C              ENDIF
C
C              Work  on requesting Northern Hemisphere data
C              Call GGRD to place grid parameters into a common field
C              Call VLLXY to convert latitude and longitude into x and y
C              Call PFLDID to place ISIS calling parameters into a
C                     common field
C              Call GETFLD to retrieve ice data from ISIS
C              If GETFLD was unsuccessful then
C                Call DTGMOD to modify date to 12 hours back
C                Increment the NCOUNT
C                IF NCOUNT equals 3 then
C                  Print No ice available message
C                  Go to End of ice requests
C                ENDIF
C                GO TO GGRD
C              ENDIF
C
C              Call FINTRP to interpolate ISIS data to wind data points
C              Combine northern and southern ice data
C              Check each ice point for coverage
C              If ice coverage is less than 55 percent THEN
C                Increment the j counter
C                Save lat and lon of the point
C                Save the specific data element in the data array
C              Endif
C              Move j to ni
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE.....................................
C
   implicit none
C
   integer n, m
   parameter(n=25000, m=33)
   integer data(n,m), nd
   integer geom1, geom2, vstat1, vstat2
```

```
      character unitll
      integer ifld1, ifld2, istgr, iwrp
      integer stat, stat1, stat2
      integer lvlcnt, ilen1, ilen2
      parameter(lvlcnt=1, ilen1=900*91, ilen2=900*136)
      real lvl1, lvl2, zlvl, fcstper, fld1(ilen1), fld2(ilen2)
      character*32 mdltype, geomnm1, geomnm2, parmnm, units
      character*24 dsetnm, lvltype
      character*200 pathnm
      character*8 suffix, seclevl
      character*16 dtg
      character*56 remark
      character*80 title
      real lat(n), lon(n), x(n), y(n)
      real shem(n), nhem(n), ice(n)
      integer mwrk, min, nin1, nin2, iflagi
      real fvali, fvalo, filval
      integer i, mod, j, k, scount, ncount, ni
      character*10 dtgn12, sdtg, ndtg
C
      geom1=1
      geom2=2
      unitll='d'
C
      ifld1=1
      ifld2=2
C
      mdltype='MISC_GRIDS'
      dsetnm='anal_ops'
      pathnm=' '
      suffix=' '
      lvltype='dpth_sfc'
      geomnm1='s_hem_900x91'
      geomnm2='n_hem_900x136'
      parmnm='ice_cvrg'
      units='percent'
      istgr=0
      iwrp=1
      lvl1=0.0
      lvl2=0.0
      fcstper=0.0
C
      mwrk=900
      min=900
      nin1=91
      nin2=136
      iflagi=1
      fvali=1.0e10
      fvalo=1.0e10
      filval=1.0e10
C
      sdtg=dtg
```

```
     ndtg=dtg
     mod=-12
C
     scount=0
     ncount=0
C
C ------------------------ For Southern Hemisphere ------------------------
C  Puts grid parameters into common field and designates geom1 as the pointer.
C
     call ggrd(geomnm1,geom1,vstat1)
     if (vstat1.lt.0) go to 3000                ! Check for error
C
C  Converts latitudes and longitudes to x and y.
C
     call vllxy(geom1,nd,lat,lon,unitll,x,y,vstat2)
     if (vstat2.eq.-1) go to 3000               ! Check for error
C
C  Look for most recent analysis.
C
 1000 continue
C
C  Puts data parameters into common fields and designates ifld1 as the pointer.
C
     call pfldid(ifld1,mdltype,dsetnm,pathnm,suffix,lvltype, geomnm1,parmnm,units,
    + remark,title,seclevl,sdtg,istgr, iwrp,lvl1,lvl2,lvlcnt,zlvl,fcstper,stat1)
     if (stat1.eq.-1) go to 3000                ! Check for error
C
C  Uses ifld1 to call data from the ISIS database.
C
     call getfld(ifld1,fld1,ilen1,stat2)
C
C  Check for successful get.
C
     if (stat2.ne.0) then
       call dtgmod(sdtg,mod,dtgn12,stat)
       sdtg=dtgn12
       scount=scount+1
       if (scount.eq.5) then
         write(0,*) 'FATAL ERROR: No ice data available (Southern Hemisphere)'
         go to 3000
       endif
       go to 1000
     endif
 1100 continue
C
C  The isis data is then interpolated to the wind data measurement points.
C
     call fintrp(x,y,nd,fld1,mwrk,min,nin1,iflagi,fvali,fvalo, filval,shem)
C
C ------------------------ For Northern Hemisphere ------------------------
C  Puts grid parameters into common field and designates geom2 as the pointer.
C
```

```
      call ggrd(geomnm2,geom2,vstat1)
      if (vstat1.lt.0) go to 3000            ! Check for error
C
C Converts latitudes and longitudes to x and y.
C
      call vllxy(geom2,nd,lat,lon,unitll,x,y,vstat2)
      if (vstat2.eq.-1) go to 3000            ! Check for error
C
C Look for most recent analysis.
C
 2000 continue
C
C Puts data parameters into common fields and designates ifld2 as the pointer.
C
      call pfldid(ifld2,mdltype,dsetnm,pathnm,suffix,lvltype, geomnm2,parmnm,units,
     + remark,title,seclevl,ndtg,istgr, iwrp,lvl1,lvl2,lvlcnt,zlvl,fcstper,stat1)
      if (stat1.eq.-1) go to 3000            ! Check for error
C
C Uses ifld2 to call data from the ISIS database.
C
      call getfld(ifld2,fld2,ilen2,stat2)
C
C Check for successful get.
C
      if (stat2.ne.0) then
        call dtgmod(ndtg,mod,dtgn12,stat)
        ndtg=dtgn12
        ncount=ncount+1
        if (ncount.eq.5) then
          write(0,*) 'FATAL ERROR: No ice data available (Northern Hemisphere)'
          go to 3000
        endif
        go to 2000
      endif
 2100 continue
C
C The isis data is then interpolated to the wind data measurement points.
C
      call fintrp(x,y,nd,fld2,mwrk,min,nin2,iflagi,fvali,fvalo, filval,nhem)
C
C ---------------------- Combine Southern and Northern ----------------------
C Values of 1e10 are assigned for points off the ice grids. The Northern Hemisphere
C grid covers 36-90 deg N (-180-180) and the Southern Hemisphere grid covers
C 54-90 deg S (-180-180). Therefore, the ice values for the latitudes between the grids
C must be given values of 0%.
C
      do 100 i=1,nd
        if (shem(i).eq.1.0e10) then
          ice(i)=nhem(i)
        else
          ice(i)=shem(i)
        endif
        if (lat(i).lt.36.0 .and. lat(i).gt.-54.0) ice(i)=0.0
 100 continue
C
C  Quality check ice data for values less than 55% and only return data not over ice in
C array data.
C
     j=0
     do 200 i=1,nd
       if (ice(i).lt.55.0) then
        j=j+1
        lat(j)=lat(i)
        lon(j)=lon(i)
        do 210 k=1,32
          data(j,k)=data(i,k)
 210      continue
       endif
 200 continue
     ni=j
C
     return                    ! Successful completion
 3000 continue
     call dbstop()              ! Close access to ISIS database
     call EXIT(ICE_ERR)
    END
```

```
        SUBROUTINE  WINDDATA(LAT,LON,ND,DTG,WSPD,WDIR)
      use CONSTANTS
C
C MODULE NAME:      WINDDATA
C
C DESCRIPTION:
C      Reads global wind speed and direction and interpolates to the desired points.
C
C USAGE:    CALL WINDDATA(LAT, LON, ND, DTG, WSPD, WDIR)
C
C PARAMETERS:
C   NAME    TYPE        USAGE       DESCRIPTION
C   --------  --------    ----------  -------------------
C   LAT     REAL        IN          Array of latitude points
C   LON     REAL        IN          Array of longitude points
C   ND      INT         IN          Number of wind directions
C   DTG     CHAR        IN          Date time group
C   WSPD    REAL        OUT         Array of wind speeds
C   WDIR    REAL        OUT         Array of wind directions
C
C MODULES CALLED:
C
C NAME     DESCRIPTION
C --------  -------------------
C GGRD     Puts grid parameters into common field
C VLLXY    Converts latitudes and longitudes to x and y
C PFLDID   Puts data parameters into common fields
C GETFLD   Calls data from the ISIS database
C DTGMOD   Modifies the date time group
C FINTRP   Interpolates data
C CCTOPC   Changes U/V components to speed and direction
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME     TYPE       DESCRIPTION
C --------  -------    -----------------
C N        INT        Max expected # of records
C GEOM     INT        Pointer
C VSTAT1   INT        Status of return
C VSTAT2   INT        Status of return
C UNIT11   CHAR       Units of lat and lon for use in VLLXY
C IFLD1    INT        Pointer for PFLDID
C IFLD2    INT        Pointer for PFLDID
C ISTGR    INT        Set to 0 if field is scaler
C IWRP     INT        Set to 1 if field is periodic, 0 for regional map
C LVLCNT   INT        Level count
C ILEN     INT        Number of points to be interpolated
C STAT1    INT        Status of return
C STAT2    INT        Status of return
C STAT3    INT        Status of return
C STAT4    INT        Status of return
C LVL1     FLOAT      First level of ISIS field parameter
```

```
C LVL2      FLOAT     Second level of ISIS field parameter
C ZLVL      FLOAT     Array containing the depth/height values for each lvlcnt
C FCSTPER   FLOAT     Forcast period
C FLD1      FLOAT     Array filled with grid data
C FLD2      FLOAT     Array filled with grid data
C MDLTYPE   CHAR      Model type
C GEOMNM    CHAR      GEOM name
C PARMNM    CHAR      Parameter name
C UNITS     CHAR      Data units requested
C DSETNM    CHAR      Data set requested
C LVLTYPE   CHAR      Level type
C PATHNM    CHAR      Path name
C SUFFIX    CHAR      not used
C SECLEVL   CHAR      Security level
C DTG       CHAR      Date time group
C REMARK    CHAR      not used
C TITLE     CHAR      not used
C X         FLOAT     Array of X values converted from latitudes
C Y         FLOAT     Array of Y values converted from longitudes
C YCOMP     FLOAT     Array of Y values
C UCOMP     FLOAT     Array of U values
C MWRK      INT       Actual first dimension of ice grid (normally = min)
C MIN       INT       First dimension of grid
C NIN       INT       Second dimension of grid
C IFLAGI    INT       Flag specifying if input fields may contain undefined values
C                       0 = no,  1 = yes
C FVALSD    INT       Value in output to specify undefined points
C FVALI     FLOAT     Value expected in input to specify undefined points
C FVALO     FLOAT     Value in output to specify undefined points
C FILVAL    FLOAT     Value in output to specify field off the input grid
C DEG       CHAR      Degree conversion flag
C DTGN6     CHAR      DTG with 6 hours added
C I         INT       Loop parameters
C MOD       INT       DTG modifier
C COUNT     INT       Number of ISIS records read
C DTGSTAT   INT       Status of DTGMOD call
C WIND_ERRINT         EXIT code from CONSTANTS file
C
C METHOD:
C       Initialize variables
C       For WIND U Component
C               Perform GGRD to put grid parameters into common field
C               Perform VLLXY to convert latitudes and longitudes to x and y
C               Perform PFLDID to put data parameters into common fields
C               Perform GETFLD to call data from the ISIS database
C       For WIND U Component
C               Perform PFLDID to put data parameters into common fields
C               Perform GETFLD to call data from the ISIS database
C       If either the u comp or the v comp is not available at the initial
C       dtg, dsetnm, and forcast period, then increment the dtg back 6
C       hours, change the dsetnm from analysis to forcast and increment the
C       forcast by 6 hours.
```

```
C       Perform FINTRP to Interpolate the wind components
C       Perform CCTOPC to change U/V components to speed and direction
C       Convert wind direction to 0 to 360 deg.
C
C  COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C..............END   PROLOGUE.....................................
C
      implicit none
C
      integer n
      parameter(n=25000)
      integer nd, geom, vstat1, vstat2
      character unitll
      integer ifld1, ifld2, istgr, iwrp, lvlcnt, ilen
      parameter(lvlcnt=1, ilen=360*181)
      integer stat1, stat2, stat3, stat4
      real lvl1, lvl2, zlvl, fcstper, fld1(ilen), fld2(ilen)
      character*32 mdltype, geomnm, parmnm, parmnm2, units
      character*24 dsetnm, lvltype
      character*200 pathnm
      character*8 suffix, seclevl
      character*16 dtg
      character*56 remark
      character*80 title
      real lat(n), lon(n), x(n), y(n)
      real ucomp(n), vcomp(n), wspd(n), wdir(n)
      integer mwrk, min, nin, iflagi, fvalsd
      real fvali, fvalo, filval
      character deg, dtgn6*10
      integer mod, count, dtgstat
      integer i
C
      geom=1
      unitll='d'
C
      ifld1=3
      ifld2=4
C
      mdltype='NOGAPS'
      dsetnm='anal_ops'
      pathnm=' '
      suffix=' '
      lvltype='ht_sfc'
      geomnm='global_360x181'
      parmnm='wnd_ucmp'
      parmnm2='wnd_vcmp'
      units='m/s'
      istgr=0
      iwrp=1
      lvl1=10.0
      lvl2=0.0
      fcstper=0.0
C
      mwrk=360
      min=360
      nin=181
      iflagi=1
      fvali=1.0e10
      fvalo=1.0e10
      filval=1.0e10
C
      mod=-6
C
      count=0
C
C  Puts grid parameters into common field and designates geom as the pointer.
C
      call ggrd(geomnm,geom,vstat1)
      if (vstat1.lt.0) go to 1100                ! Check for error
C
C  Converts latitudes and longitudes to x and y.
C
      call vllxy(geom,nd,lat,lon,unitll,x,y,vstat2)
      if (vstat2.eq.-1) go to 1100               ! Check for error
C
 1000 continue
C
C ---------------------- For Wind U Component ------------------------
C  Puts data parameters into common fields and designates ifld1 as the pointer.
C
      call pfldid(ifld1,mdltype,dsetnm,pathnm,suffix,lvltype,geomnm, parmnm,units,
     + remark,title,seclevl,dtg,istgr,iwrp,lvl1,lvl2, lvlcnt,zlvl,fcstper,stat1)
      if (stat1.eq.-1) go to 1100                ! Check for error
C
C  Uses ifld1 to call data from the ISIS database.
C
      call getfld(ifld1,fld1,ilen,stat2)
C
C ---------------------- For Wind V Component ----------------------
C  Puts data parameters into common fields and designates ifld2 as the pointer.
C
      call pfldid(ifld2,mdltype,dsetnm,pathnm,suffix,lvltype,geomnm, parmnm2,units,
     + remark,title,seclevl,dtg,istgr,iwrp,lvl1,lvl2, lvlcnt,zlvl,fcstper,stat3)
      if (stat3.eq.-1) go to 1100                ! Check for error
C
C  Uses ifld2 to call data from the ISIS database.
C
      call getfld(ifld2,fld2,ilen,stat4)
C
C ------------ Check for successful 'get' of ISIS data -------------
C  If either the u comp or the v comp is not available at the initial dtg, dsetnm, and forcast
C period, then increment the dtg back 6 hours, change the dsetnm from analysis to forcast
C and increment the forcast by 6 hours.
```

```
C
      if (stat2.ne.0 .or. stat4.ne.0) then
        count=count+1
        call dtgmod(dtg,mod,dtgn6,dtgstat)
        dtg=dtgn6
         dsetnm='fcst_ops'
        fcstper=fcstper + 6.0
        if (count.eq.5) then
          write(0,*) 'FATAL ERROR: No NOGAPS wind data available'
          go to 1100
        endif
        go to 1000
      endif
C
C ------------------- Interpolate wind components ------------------
C  The isis data is then interpolated to the scatterometer data measurement points.
C
C  U component:
C
      call fintrp(x,y,nd,fld1,mwrk,min,nin,iflagi,fvali,fvalo, filval,ucomp)
C
C  V component:
C
      call fintrp(x,y,nd,fld2,mwrk,min,nin,iflagi,fvali,fvalo, filval,vcomp)
C
C  Changes U/V components to speed and direction.
C
       deg='d'
      fvalsd=10000000000
      call cctopc(ucomp,vcomp,nd,deg,iflagi,fvalsd,wdir,wspd)
C
C  Convert wind direction from 0 to 180 deg and 0 to -180 dtg to 0 to 360 deg and
C change convention from 'direction toward' to 'direction from'.
C
      do 100 i=1,nd
        wdir(i)=wdir(i)+180
 100 continue
C
      return                    ! Successful completion
 1100 continue
      call dbstop()             ! Close access to ISIS database
      call EXIT(WIND_ERR)
      END
```

```
SUBROUTINE   CONTIN(FDATA,NF,IBSAVE,NR,CONT)
C
C MODULE NAME:      CONTIN
C
C DESCRIPTION:
C      This subroutine checks that the filtered blocks are continguous, i.e. within 1.5
C      minutes of each other, and assigns a value of 1 to the flag 'cont' for the block if
C      true.  'cont'= 0 begins a continuous set of blocks. Also, the number of records per
C      block are saved in the array 'nr'.
C
C USAGE:    CALL CONTIN(FDATA, NF, IBSAVE, NR, CONT)
C
C PARAMETERS:
C   NAME    TYPE          USAGE        DESCRIPTION
C   --------  --------        ----------        --------------------
C FDATA    INT           IN           Array of filtered data blocks
C NF       INT           N            Number of elements in FDATA
C IBSAVE   INT           OUT          Saved number of blocks
C NR       INT           OUT          Array of saved records in each block
C CONT     INT           OUT          Contiguous flag array
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME       TYPE   DESCRIPTION
C ----      ----   -----------
C N          INT    Max expected # of records
C M          INT    Number of fields in each record
C B          INT    Max expected # of blocks
C BLK               INT    Block Number
C IND        INT    Number of blocks
C SS1        INT    Old block time
C SS2        INT    New block time
C DY1               INT    Old block day
C DY2               INT    New block day
C I, J, K    INT    Loop variables
C R          INT    Records in a block counter
C
C METHOD:
C      Initialize variables
C      Loop on number of data elements
C        If this is a new block then
C          Save the number of records in the block
C          Increment the block number counter
C          Compute the time
C          Set the continuity flag
C          Reset time
C          Reset block number
C        Endif
C        Increment block record counter
C      End loop
C      Return number of blocks
C
```

```
C  COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE....................................
C
      implicit none
C
      integer n, m, b
      parameter(n=25000, m=33, b=70)
      integer fdata(n,m), nf
      integer ibsave, nr(b), cont(b)
       integer blk, ind, ss1, ss2, dy1, dy2, dif
      integer i, j, k, r
C
      r=0
      ind=1
      blk=fdata(1,32)
      ss1=fdata(1,7)*3600 + fdata(1,8)*60 + fdata(1,9)/1000
      dy1=fdata(1,6)
      cont(1)=0
C
      do 100 i=1,nf
        if (fdata(i,32).ne.blk) then          ! New block
          nr(ind)=r                          ! Saves # of records/block
          ind=ind+1                          ! Counts # of blocks
          ss2=fdata(i,7)*3600 + fdata(i,8)*60 + fdata(i,9)/1000
          dy2=fdata(i,6)
          if (dy1.lt.dy2) ss2=ss2 + 3600*24
          dif=ss2-ss1
          if (dif.le.90) then                ! Checks for continuity
            cont(ind)=1                      ! Sets flag value
          else
            cont(ind)=0
          endif
          ss1=ss2                            ! Resets block time
          dy1=dy2
          blk=fdata(i,32)                     ! Resets block number
          r=0                               ! Resets record counter
        endif
        r=r+1                                ! Counts # of records/block
 100  continue
      nr(ind)=r                         ! Saves # of records for last block
      ibsave=ind                             ! Saves # of blocks
C
      END
      SUBROUTINE PROCESS(FDATA,WSPD,WDIR,IBSAVE,NR,NF,CONT,MSG)
       use STATUS
       use CONSTANTS
        include 'SCTR_ERS.H'
C
C..............START   PROLOGUE...................................
C MODULE NAME:       PROCESS
C
C DESCRIPTION:
C        This subroutine formats the data array for processing and processes the data
C        to produce wind speed and direction from the sigma naught measurements. Then
C        the scatterometer wind directions are dealiased wrt NOGAPS wind direction.
C        The data is then passed through filters to check continuity.
C
C USAGE:    CALL PROCESS(FDATA, WSPD, WDIR, IBSAVE, NR, NF, CONT)
C
C PARAMETERS:
C   NAME     TYPE          USAGE        DESCRIPTION
C   ---------  --------      ----------   -------------------
C FDATA     INT           IN           Field data
C WSPD      REAL          IN           Wind speed
C WDIR      REAL          IN           Wind direction
C IBSAVE    INT           IN           Total number of blocks saved
C NR        INT           IN           Number of records in a block
C NF        INT           IN           Number of total records
C CONT      INT           IN           Confidence flags
C MSG       INT           OUT          Message EXIT code
C
C MODULES CALLED:
C
C NAME       DESCRIPTION
C --------   -------------------
C WINRET   Implementation of CMOD4 to process the scatterometer data
C EXTFIT   Compares model wind fields with 1st guess scatterometer solution
C SLICE    Filters data
C WRITE_LLT       Writes an llt data structure to ISIS
C
C LOCAL VARIABLES AND STRUCTURES:
C
C NAME       TYPE  DESCRIPTION
C --------   ------- ------------------
C N          INT    Max expected # of records
C M          INT    Number of fields in each record
C P          INT    Number of records in data block
C P1         INT    Number of records in data block + 1
C B          INT    Max expected # of blocks
C MM         INT    Number of antennas - 1
C RC         INT    Number of rows and cells
C GS         INT    Max number of vector solutions
C MXRW       INT    Used for SLICE (# blocks) * (# records/block)
C I,J,K      INT    LOOP VARIABLES
C ICONF      INT    CONFIDENCE FLAGS
```

```
C IBC       INT   ARRAY OF BLOCK NUMBERS
C NPC       INT   ARRAY OF TOTAL RECORDS IN A BLOCK
C PCTRK     INT   SATELLITE TRACK
C PCDM      INT   PRODUCT CONFIDENCE FLAG
C IBB       INT   ORIGINAL BLOCK ID
C IROW      INT   ROW ID
C JCELL     INT   CELL ID
C PCLT      REAL  LATITUDE
C PCLN      REAL  LONGITUDE
C VE        REAL  ESA WIND SPEED
C DE        REAL  ESA WIND DIRECTION
C VM        REAL  NOGAPS MODEL WIND SPEED
C DM        REAL  NOGAPS MODEL WIND DIRECTION
C AI        REAL  RADAR INCIDENCE ANGLE
C BA        REAL  RADAR AZMUTH ANGLE
C S0        REAL  BACK SCATTER
C KP        REAL  NOISE PERCENT
C MS        REAL  MISSING PACKET COUNTER
C NS4       INT   ARRAY OF VALID BLOCK NUMBERS
C IYMD      INT   ARRAY OF YEAR, MONTH, DAYS
C IHMS      INT   ARRAY OF HOUR, MINUTE, SECONDS
C NVS       INT   VALID BLOCK NUMBER
C VS1       REAL  Wind speed solution out of EXTFIT
C DS1       REAL  Wind direction solution out of EXTFIT
C RS1       REAL  Solution residual out of EXTFIT
C VSOL      REAL  Wind speed solution out of WINRET
C DSOL      REAL  Wind direction solution out of WINRET
C RSOL      REAL  Solution residual out of WINRET
C YS0       REAL  BACKSCATTER
C YAI       REAL  RADAR INCIDENCE ANGLE
C YBA       REAL  RADAR AZMUTHUAL ANGLE
C YKP       REAL  NOISE PERCENT
C YMS       REAL  MISSING PACKET COUNTER
C YPCLT     REAL  POINT LATITUDE
C YPCLN     REAL  POINT LONGITUDE
C YVM       REAL  NOGAPS MODEL WIND SPEED
C YDM       REAL  NOGAPS MODEL WIND DIRECTION
C VSX       REAL  PROBABLE WIND SPEED SOLUTIONS
C DSX       REAL  PROBABLE WIND DIRECTION SOLUTIONS
C RSX       REAL  SOLUTION PROBABLY
C DFX       REAL  AVERAGE DIRECTION
C IBCX      INT   ARRAY OF BLOCK NUMBERS
C NPCX      INT   ARRAY OF TOTAL RECORDS IN A BLOCK
C ICONFX    INT   CONFIDENCE FLAGS
C IROWX     INT   ROW ID
C JCELLX    INT   CELL ID
C PCTRKX    INT   SATELLITE TRACK
C NS4X      INT   ARRAY OF VALID BLOCK NUMBERS
C IYMDX     INT   ARRAY OF YEAR, MONTH, DAYS
C IHMSX     INT   ARRAY OF HOUR, MINUTE, SECONDS
C PCLTX     REAL  POINT LATITUDE
C PCLNX     REAL  POINT LONGITUDE
```

```
C  VEX        REAL  ESA WIND SPEED
C  DEX        REAL  ESA WIND DIRECTION
C  VMX        REAL  NOGAPS MODEL WIND SPEED
C  DMX        REAL  NOGAPS MODEL WIND DIRECTION
C  BAX        REAL  SATELLITE LOOK ANGLE
C  AIX        REAL  RADAR INCIDENCE ANGLE
C  BAXX       REAL  RADAR AZMUTHUAL ANGLE
C  S0X        REAL  BACKSCATTER
C  KPX        REAL  NOISE PERCENT
C  MSX        REAL  MISSING PACKET COUNTER
C  PCDMX      REAL  UWI PRODUCT CONFIDENCE FLAG
C  IPCDX      INT   PRODUCT CONFIDENCE FLAG
C  IBEGIN     INT   FIRST BLOCK OF CONTIGUOUS BLOCK SET
C  ILAST      INT   LAST BLOCK OF CONTIGUOUS BLOCK SET
C  NROW       INT   NUMBER OF ROWS SAVED
C  IBCY       INT   ARRAY OF BLOCK NUMBERS
C  NPCY       INT   ARRAY OF TOTAL RECORDS IN A BLOCK
C  ICONFY     INT   CONFIDENCE FLAGS
C  IROWY      INT   ROW ID
C  JCELLY     INT   CELL ID
C  PCTRKY     INT   SATELLITE TRACK
C  NS4Y       INT   ARRAY OF VALID BLOCK NUMBERS
C  IYMDY      INT   ARRAY OF YEAR, MONTH, DAYS
C  IHMSY      INT   ARRAY OF HOUR, MINUTE, SECONDS
C  PCLTY      REAL  POINT LATITUDE
C  PCLNY      REAL  POINT LONGITUDE
C  VEY        REAL  ESA WIND SPEED
C  DEY        REAL  ESA WIND DIRECTION
C  VMY        REAL  NOGAPS MODEL WIND SPEED
C  DMY        REAL  NOGAPS MODEL WIND DIRECTION
C  BAY        REAL  SATELLITE LOOK ANGLE
C  IPCD       INT   PRODUCT CONFIDENCE FLAG
C  VS3        REAL  PROBABLE WIND SPEED SOLUTIONS
C  DS3        REAL  PROBABLE WIND DIRECTION SOLUTIONS
C  RS3        REAL  SOLUTION PROBABLY
C  DF         REAL  AVERAGE DIRECTION
C  AIY        REAL  RADAR INCIDENCE ANGLE
C  BAYY       REAL  RADAR AZMUTHUAL ANGLE
C  S0Y        REAL  BACKSCATTER
C  KPY        REAL  NOISE PERCENT
C  MSY        REAL  MISSING PACKET COUNTER
C  PCDMY      REAL  UWI PRODUCT CONFIDENCE FLAG
C  LATITUDEREAL  LATITUDE
C  LONGITUDE        REAL  LONGITUDE
C  IYR        INT   YEAR
C  IMO        INT   MONTH
C  IDAY       INT   DAY
C  IMIN       INT   MINUTE
C  ISEC       INT   SECOND
C  CLOSE_ERR        INT   EXIT code from CONSTANTS file
C
C  METHOD:
```

```
C       Fill dummy arrays with bogus values. Move values from array DATA into variables
C       and arrays. Move values from variables into single variables for WINRET.
C       Move values from WINRET into arrays
C       Perform EXTFIT
C       Overwrite dummy array values with real data
C       Set up contiguous blocks of arrays for SLICE
C       Perform SLICE
C       Place data into ISIS structure
C       Write data out into a file
C
C COMPILER DEPENDENCES:  FORTRAN 90 - EDINBURGH
C
C.............END   PROLOGUE.....................................
C
C Note: 'implicit none' is not used due to the large number of counter variables.
C
       integer n, m, p, p1, b, mm, rc, gs, mxrw
       parameter (n=25000, m=33, p=361, p1=p+1, b=70, mm=2, rc=19)
      parameter (gs=4, mxrw=rc*b)
      integer fdata(n,m), nr(b), cont(b)
      integer ibsave, nf, msg
      integer i, j, k
      integer iconf(b), ibc(b), npc(b)
      real wspd(n), wdir(n)
C
       integer sid(p1,b), pctrk(p1,b), iid(p1,b), yr(p1,b), mo(p1,b), dy(p1,b), hr(p1,b),
     + min(p1,b),sec(p1,b), pcdm(p1,b), ibb(p1,b), irow(p1,b), jcell(p1,b)
       real pclt(p1,b), pcln(p1,b), ve(p1,b), de(p1,b), vm(p1,b), dm(p1,b), ai(0:mm,p1,b),
     + ba(0:mm,p1,b), s0(0:mm,p1,b), kp(0:mm,p1,b), ms(0:mm,p1,b)
C
      integer ns4(p1,b),iymd(p1,b),ihms(p1,b)
     integer iypcdm,nvs
       real vs1(gs,p1,b),ds1(gs,p1,b),rs1(gs,p1,b)
     real vsol(1:gs),dsol(1:gs),rsol(1:gs)

     real ys0(0:mm),yai(0:mm),yba(0:mm),ykp(0:mm),yms(0:mm),
    +     ypclt,ypcln,yvm,ydm
C
C ---------- Array variables to account for missing data. ----------
      integer sidx(rc,rc,b), pctrkx(rc,rc,b), iidx(rc,rc,b), ibymdx(rc,rc,b), ibhmsx(rc,rc,b),
     + ibcx(rc,rc,b), irowx(rc,rc,b), jcellx(rc,rc,b), npcx(rc,rc,b), iconfx(rc,rc,b),
     + ns4x(rc,rc,b),pcdmx(rc,rc,b)
      real pcltx(rc,rc,b), pclnx(rc,rc,b), vex(rc,rc,b), dex(rc,rc,b), vmx(rc,rc,b),
     + dmx(rc,rc,b),aix(0:mm,rc,rc,b), bax(0:mm,rc,rc,b), s0x(0:mm,rc,rc,b),
     + kpx(0:mm,rc,rc,b),msx(0:mm,rc,rc,b)
C
     integer*2 IPCDx(rc,rc,b)
      real VSx(gs,rc,rc,b),DSx(gs,rc,rc,b)
      real RSx(gs,rc,rc,b),DFx(rc,rc,b)
C
C ------------------- Variables used with SLICE. -------------------
integer ibegin(b)
```

```
      integer ilast(b)
      integer nrow(mxrw)
C
      integer sidy(rc,mxrw), pctrky(rc,mxrw), iidy(rc,mxrw),
     +      ibymdy(rc,mxrw), ibhmsy(rc,mxrw), ibcy(rc,mxrw),
     +      irowy(rc,mxrw), jcelly(rc,mxrw), npcy(rc,mxrw),
     +      iconfy(rc,mxrw), ns4y(rc,mxrw), pcdmy(rc,mxrw)
      real pclty(rc,mxrw), pclny(rc,mxrw), vey(rc,mxrw), dey(rc,mxrw),
     +      vmy(rc,mxrw), dmy(rc,mxrw), aiy(0:mm,rc,mxrw),
     +      bay(0:mm,rc,mxrw), s0y(0:mm,rc,mxrw), kpy(0:mm,rc,mxrw),
     +      msy(0:mm,rc,mxrw)
C
      integer*2 ipcd(rc,mxrw)
       real vs3(gs,rc,mxrw),ds3(gs,rc,mxrw),rs3(gs,rc,mxrw)
      real df(rc,mxrw)
C
C ------------- Variables for writing to ISIS ---------------
C
      real :: latitude                    ! -90. to 90.
      real :: longitude                   ! -180. to nearly 180.
      integer :: iyr, imo, iday, ihr, imin, isec
      integer :: istat
      type(sctr_ers) :: sctr              ! Data block read from file
C
C ----------------------- Format data for processing ----------------------
C
      k=0
      do 100 i=1,ibsave
        iconf(i)=cont(i)
        ibc(i)=i
        npc(i)=nr(i)
C
        do 110 j=1,npc(i)
          k=k+1
          sid(j,i)=fdata(k,1)
          pctrk(j,i)=fdata(k,2)
          iid(j,i)=fdata(k,3)
          yr(j,i)=fdata(k,4)
          mo(j,i)=fdata(k,5)
          dy(j,i)=fdata(k,6)
          hr(j,i)=fdata(k,7)
          min(j,i)=fdata(k,8)
          sec(j,i)=fdata(k,9)/1000
          pclt(j,i)=float(fdata(k,10))/100.
          pcln(j,i)=float(fdata(k,11))/100.
          ai(0,j,i)=float(fdata(k,12))/10.
          ba(0,j,i)=float(fdata(k,13))/10.
          s0(0,j,i)=float(fdata(k,14))/100.
          kp(0,j,i)=float(fdata(k,15))/10.
          ms(0,j,i)=float(fdata(k,16))
          ai(1,j,i)=float(fdata(k,17))/10.
          ba(1,j,i)=float(fdata(k,18))/10.
```

```
            s0(1,j,i)=float(fdata(k,19))/100.
            kp(1,j,i)=float(fdata(k,20))/10.
            ms(1,j,i)=float(fdata(k,21))
            ai(2,j,i)=float(fdata(k,22))/10.
            ba(2,j,i)=float(fdata(k,23))/10.
            s0(2,j,i)=float(fdata(k,24))/100.
            kp(2,j,i)=float(fdata(k,25))/10.
            ms(2,j,i)=float(fdata(k,26))
            ve(j,i)=float(fdata(k,27))/10.
            de(j,i)=float(fdata(k,28))
            pcdm(j,i)=fdata(k,29)
            irow(j,i)=fdata(k,30)
            jcell(j,i)=fdata(k,31)
            ibb(j,i)=fdata(k,32)
            vm(j,i)=wspd(k)
            dm(j,i)=wdir(k)
           iymd(j,i)=(yr(j,i)-yr(j,i)/100*100)*10000+mo(j,i)*100+dy(j,i)
            ihms(j,i)=hr(j,i)*10000+min(j,i)*100+sec(j,i)
  110   continue
  100 continue
C
C -----------------------------------------------------------------------
C                         Begin Processing
C -----------------------------------------------------------------------
C  This portion of the subroutine processes the ERS-1 scatterometer s0's using NCEP's
C processing scheme.  It performs the processing record by record.
C
do 200 ibsc = 1, ibsave                    ! total # blocks
       do 210 iii = 1, npc(ibsc)               ! # records in block
          do jjj = 0,2
            ys0(jjj) = s0(jjj,iii,ibsc)
            yai(jjj) = ai(jjj,iii,ibsc)
            yba(jjj) = ba(jjj,iii,ibsc)
            ykp(jjj) = kp(jjj,iii,ibsc)
            yms(jjj) = ms(jjj,iii,ibsc)
          enddo
          iypcdm = 0
          ypclt = pclt(iii,ibsc)
          ypcln = pcln(iii,ibsc)
          yvm = vm(iii,ibsc)
          ydm = dm(iii,ibsc)
C
C ------------------------------ Inversion ------------------------------
call winret(ys0,yai,yba,ykp,yms,iypcdm,vsol,dsol,rsol,nvs)
C
          ns4(iii,ibsc) = nvs
C
C -------------------------- Ambiguity Removal --------------------------
call extfit(ypclt,ypcln,iypcdm,vsol,dsol,rsol,yvm,ydm)
C
          do kk = 1, 4
            vs1(kk,iii,ibsc) = vsol(kk)
            ds1(kk,iii,ibsc) = dsol(kk)
            rs1(kk,iii,ibsc) = rsol(kk)
          enddo
 210    continue
 200 continue
C
C ------------ Filling in dummy array to account for missing data -------------
C  The dummy array is filled with bogus values to avoid discontinuities.
C
     do 300 i2b = 1,ibsave
       do 310 i2r = 1,rc
         do 320 i2c = 1,rc
           do i2n = 1,gs
             VSx(i2n,i2c,i2r,i2b) = 51.0
             DSx(i2n,i2c,i2r,i2b) = 510.0
             RSx(i2n,i2c,i2r,i2b) = 0.0
           enddo
C
           do i2n = 0,mm
             aix(i2n,i2c,i2r,i2b) = 0.0
             bax(i2n,i2c,i2r,i2b) = 0.0
             s0x(i2n,i2c,i2r,i2b) = 0.0
             kpx(i2n,i2c,i2r,i2b) = 0.0
             msx(i2n,i2c,i2r,i2b) = 0.0
           enddo
C
            sidx(i2c,i2r,i2b) = 0
           pctrkx(i2c,i2r,i2b) = 510
            iidx(i2c,i2r,i2b) = 0
           ibymdx(i2c,i2r,i2b) = 0
           ibhmsx(i2c,i2r,i2b) = 0
            pcltx(i2c,i2r,i2b) = 100.0
            pclnx(i2c,i2r,i2b) = 510.0
            pcdmx(i2c,i2r,i2b) = 0
             ibcx(i2c,i2r,i2b) = i2b
             npcx(i2c,i2r,i2b) = npc(i2b)
            irowx(i2c,i2r,i2b) = i2r
           jcellx(i2c,i2r,i2b) = i2c
           iconfx(i2c,i2r,i2b) = iconf(i2b)
              vex(i2c,i2r,i2b) = 51.0
              dex(i2c,i2r,i2b) = 510.0
              vmx(i2c,i2r,i2b) = 51.0
              dmx(i2c,i2r,i2b) = 510.0
             ns4x(i2c,i2r,i2b) = 0
              DFx(i2c,i2r,i2b) = 510.0
            IPCDx(i2c,i2r,i2b) = 0
 320     continue
 310   continue
 300 continue
C
C ------------ Over-write dummy array values with real data --------------
C  The bogus values are overwritten with the real data available and the
```

```
C  full block arrays are added together.
C
     do 400 i3b = 1,ibsave                  ! total # of blocks
C
       do 410 iii = 1,npc(i3b)           ! where npc = # records in block
C
          do i2n = 1,gs
        VSx(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = vs1(i2n,iii,i3b)
        DSx(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = ds1(i2n,iii,i3b)
        RSx(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = rs1(i2n,iii,i3b)
          enddo
C
          do i2n = 0,mm
        aix(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = ai(i2n,iii,i3b)
        bax(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = ba(i2n,iii,i3b)
        s0x(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = s0(i2n,iii,i3b)
        kpx(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = kp(i2n,iii,i3b)
        msx(i2n,jcell(iii,i3b),irow(iii,i3b),i3b) = ms(i2n,iii,i3b)
          enddo
C
          sidx(jcell(iii,i3b),irow(iii,i3b),i3b) = sid(iii,i3b)
        pctrkx(jcell(iii,i3b),irow(iii,i3b),i3b) = pctrk(iii,i3b)
          iidx(jcell(iii,i3b),irow(iii,i3b),i3b) = iid(iii,i3b)
        ibymdx(jcell(iii,i3b),irow(iii,i3b),i3b) = iymd(iii,i3b)
        ibhmsx(jcell(iii,i3b),irow(iii,i3b),i3b) = ihms(iii,i3b)
         pcltx(jcell(iii,i3b),irow(iii,i3b),i3b) = pclt(iii,i3b)
         pclnx(jcell(iii,i3b),irow(iii,i3b),i3b) = pcln(iii,i3b)
         pcdmx(jcell(iii,i3b),irow(iii,i3b),i3b) = pcdm(iii,i3b)
          ibcx(jcell(iii,i3b),irow(iii,i3b),i3b) = ibc(i3b)
          npcx(jcell(iii,i3b),irow(iii,i3b),i3b) = npc(i3b)
         irowx(jcell(iii,i3b),irow(iii,i3b),i3b) = irow(iii,i3b)
        jcellx(jcell(iii,i3b),irow(iii,i3b),i3b) = jcell(iii,i3b)
        iconfx(jcell(iii,i3b),irow(iii,i3b),i3b) = iconf(i3b)
           vex(jcell(iii,i3b),irow(iii,i3b),i3b) = ve(iii,i3b)
           dex(jcell(iii,i3b),irow(iii,i3b),i3b) = de(iii,i3b)
           vmx(jcell(iii,i3b),irow(iii,i3b),i3b) = vm(iii,i3b)
           dmx(jcell(iii,i3b),irow(iii,i3b),i3b) = dm(iii,i3b)
          ns4x(jcell(iii,i3b),irow(iii,i3b),i3b) = ns4(iii,i3b)
C
 410   continue
 400 continue
C
C ------------------- Set up contiguous blocks of arrays for SLICE ---------------------
C     if iconf(ibsc) = 0 then 1st block of contiguous block set, ibegin(ii)
C     if iconf(ibsc) = 1 then part of contiguous block set, last = ilast(ii)
C     if iconf(ibsc) = 0 then next block of contiguous block set, ibegin(ii+1)
C     if iconf(ibsc) = 1 then part of contiguous block set, last = ilast(ii+1)
C         ibc(ibsc) = block id
C            ibsave = total no. of blocks
C
    ii = 0
    do 500 jj = 1,ibsave
```

```
      if (iconf(jj) .eq. 0) then
        ii = ii + 1
        ibegin(ii) = ibc(jj)
        ilast(ii) = ibc(jj)
        nrow(ii) = rc                    ! rc = 19
      else
        ilast(ii) = ibc(jj)
        nrow(ii) = nrow(ii) + rc
      endif
  500 continue
      iisave = ii
C
C ------------------------- Prepare data for SLICE ------------------------
C
      isoln=0
      do 600 ibs = 1,iisave
        iii1 = ibegin(ibs)
        iii2 = ilast(ibs)
        irosav = nrow(ibs)
        jjj = 0
        do 610 iii = iii1, iii2
          do 620 ijk = 1, rc
            jjj = jjj + 1          ! counting rows from iii1 to iii2
            do 630 kkk = 1,rc          ! cell id
              do kk = 1, 4
                vs3(kk,kkk,jjj) = VSx(kk,kkk,ijk,iii)
                ds3(kk,kkk,jjj) = DSx(kk,kkk,ijk,iii)
                rs3(kk,kkk,jjj) = RSx(kk,kkk,ijk,iii)
              enddo
              do kk = 0, mm
                aiy(kk,kkk,jjj) = aix(kk,kkk,ijk,iii)
                bay(kk,kkk,jjj) = bax(kk,kkk,ijk,iii)
                s0y(kk,kkk,jjj) = s0x(kk,kkk,ijk,iii)
                kpy(kk,kkk,jjj) = kpx(kk,kkk,ijk,iii)
                msy(kk,kkk,jjj) = msx(kk,kkk,ijk,iii)
              enddo
              sidy(kkk,jjj) = sidx(kkk,ijk,iii)
            pctrky(kkk,jjj) = pctrkx(kkk,ijk,iii)
              iidy(kkk,jjj) = iidx(kkk,ijk,iii)
            ibymdy(kkk,jjj) = ibymdx(kkk,ijk,iii)
            ibhmsy(kkk,jjj) = ibhmsx(kkk,ijk,iii)
             pclty(kkk,jjj) = pcltx(kkk,ijk,iii)
             pclny(kkk,jjj) = pclnx(kkk,ijk,iii)
             pcdmy(kkk,jjj) = pcdmx(kkk,ijk,iii)
              ibcy(kkk,jjj) = ibcx(kkk,ijk,iii)
              npcy(kkk,jjj) = npcx(kkk,ijk,iii)
             irowy(kkk,jjj) = irowx(kkk,ijk,iii)
            jcelly(kkk,jjj) = jcellx(kkk,ijk,iii)
            iconfy(kkk,jjj) = iconfx(kkk,ijk,iii)
              vey(kkk,jjj) = vex(kkk,ijk,iii)
              dey(kkk,jjj) = dex(kkk,ijk,iii)
              vmy(kkk,jjj) = vmx(kkk,ijk,iii)
              dmy(kkk,jjj) = dmx(kkk,ijk,iii)
             ns4y(kkk,jjj) = ns4x(kkk,ijk,iii)
              df(kkk,jjj) = DFx(kkk,ijk,iii)
             ipcd(kkk,jjj) = IPCDx(kkk,ijk,iii)
C
 630          continue
 620        continue
 610    continue      ! End of contiguous set of data fm ibegin to ilast
C
C    $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C     SLICE  SLICE  SLICE  SLICE  SLICE  SLICE  SLICE  SLICE
C    $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
          call slice(irosav,vs3,ds3,rs3,ipcd,df)
C
C    $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C     SLICE  SLICE  SLICE  SLICE  SLICE  SLICE  SLICE  SLICE
C    $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
C  Writing out file with ambiguity removal using SLICE modal filter for contiguous set of
C  data from ibegin to ilast.
C
      do 640 jjjn = 1,irosav
        do 650 ijkn = 1,19
c ----------------------
c    sidy:          Satellite ID
c    pctrky:          Satellite track (ascending or descending orbit)
c    iidy:          Instrument ID
c    ibcy:          sequential block ID
c    ibb:           original block ID (filtered)
c    irowy:           row ID
c    jcelly:        cell ID
c    ibymdy:           Time of observation (year-month-day)
c    ibhmsy:           Time of observation (hour-min-sec)
c    pclty:         latitude
c    pclny:          longitude
c    aiy:           antenna incidence angle
c    bay:           antenna look angle
c    s0y:           backscatter
c    kpy:           noise figure
c    msy:            missing packet counter
c    vmy:            NOGAPS model wind speed
c    dmy:            NOGAPS model wind direction
c    vey:           ESA wind speed
c    dey:           ESA wind direction
c    pcdmy:            confidence flag
c    vs3(1,ijkn,jjjn): most probable wind speed solution ("chosen one")
c    ds3(1,ijkn,jjjn): most probable wind direction sol'n ("chosen one")
c    vs3(2,ijkn,jjjn): 2nd most probable wind speed sol'n
c    ds3(2,ijkn,jjjn): 2nd most probable wind dir sol'n
c    vs3(3,ijkn,jjjn): 3rd most probable wind speed sol'n
c    ds3(3,ijkn,jjjn): 3rd most probable wind dir sol'n
```

```
c    vs3(4,ijkn,jjjn): 4th most probable wind speed sol'n
c    ds3(4,ijkn,jjjn): 4th most probable wind dir sol'n
c    vmy:              NOGAPS model wind speed
c    dmy:              NOGAPS model wind direction
c -----------------------
C  Check for valid block number and number of valid solutions (ns4y) > 0.
C
           if(ns4y(ijkn,jjjn).gt.0) then
C
     isoln=isoln+1
C
C --------------- Write to ISIS ----------------
sctr%sat_num = float(sidy(ijkn,jjjn))
      sctr%sat_trak = float(pctrky(ijkn,jjjn))
      sctr%sat_inst_data_flag = float(iidy(ijkn,jjjn))
C
C  parse out the date elements
C
      iyr = ibymdy(ijkn,jjjn) / 10000
      imo = (ibymdy(ijkn,jjjn) - iyr * 10000) / 100
      iday = (ibymdy(ijkn,jjjn) - iyr * 10000 - imo * 100)
      ihr = ibhmsy(ijkn,jjjn) / 10000
      imin = (ibhmsy(ijkn,jjjn) - ihr * 10000) / 100
      isec = (ibhmsy(ijkn,jjjn) - ihr * 10000 - imin * 100)
         iyr = iyr + 1900
C
      sctr%obs_yr = float(iyr)
      sctr%obs_mo = float(imo)
      sctr%obs_day = float(iday)
      sctr%obs_hr = float(ihr)
      sctr%obs_min = float(imin)
      sctr%obs_sec = float(isec)
      sctr%crse_lat = pclty(ijkn,jjjn)
      sctr%crse_lon = pclny(ijkn,jjjn)
      latitude = pclty(ijkn,jjjn)
      longitude = pclny(ijkn,jjjn)
C
         do kk = 0,2
           sctr%radr_data(kk+1)%radr_incd_ang = aiy(kk,ijkn,jjjn)
           sctr%radr_data(kk+1)%radr_azim_ang = bay(kk,ijkn,jjjn)
           sctr%radr_data(kk+1)%bk_sctr = s0y(kk,ijkn,jjjn)
           sctr%radr_data(kk+1)%nois_pct = kpy(kk,ijkn,jjjn)
           sctr%radr_data(kk+1)%mis_pckt_cntr = msy(kk,ijkn,jjjn)
         enddo
C
      sctr%fdp_wnd_spd_10m = vey(ijkn,jjjn)
      sctr%fdp_wnd_dir_10m = dey(ijkn,jjjn)
      sctr%uwi_prod_conf_flag = pcdmy(ijkn,jjjn)
C
         do kk = 1,4
          sctr%wnd_data(kk)%wnd_spd_10m = vs3(kk,ijkn,jjjn)
          sctr%wnd_data(kk)%wnd_dir_10m = ds3(kk,ijkn,jjjn)
```

```
            sctr%wnd_data(kk)%sol_prbl = rs3(kk,ijkn,jjjn)
          enddo
C
C  Write scatter data record into the ISIS database (passing first word of ISIS structure, as
C  pointer to the entire structure). Latitude/longitude must be signed degrees, -90. to +90 /
C  -180 to +180
C
seqtype = 'sctr_ers'
            dsetnam = 'satdat'
            rsn = 'sctr_ers'
           call WRITE_LLT(sctr, latitude, longitude, iyr, imo, iday,
   *           ihr, imin, isec, err)
           if (err /= 0) then
             call dbstop()
             call EXIT(err)
           endif

           endif
 650     continue
 640   continue
C
 600 continue              ! Return to get next set of contiguous blocks
C
     call LCLOS(seqtype, vrsnnam, dsetnam, seclvl, dtg, istat)
     if (istat.lt.0) then
       write(0,*) 'MESSAGE: Error in LCLOS'
       msg=CLOSE_ERR
     endif
     return

     END
```

LIST OF REFERENCES

1. Robinson, I. S., *Satellite Oceanography*, Ellis Horwood Limited, 1985.

2. Pierson, W. J., Skylab EREP Investigation Summary, NASA SP-399, 1978.

3. Kramer, H. J., *Observation of the Earth and Its Environment*, Springer-Verlag, 1994..

4. Gemmill, W. H., Woiceshyn, P. M., Peters, C. A., and Gerald, V. M., "A Preliminary Evaluation of Scatterometer Wind Transfer Functions for ERS-1 Data," National Oceanic and Atmospheric Administration Ocean Products Center Office Note, Contribution No. 97, 1994.

5. Maul, G. A., *Introduction to Satellite Oceanography*, Martinus Nijhoff Publishers, 1985.

6. Colton, M. C., "Dependence of Radar Backscatter on the Energetics of the Air-Sea Interface," Ph.D. Thesis, Naval Postgraduate School, 1989.

7. ERS-1 Reference Manual, European Space Agency

8. Vaughan, Robin A. Microwave Remote Sensing for Oceanographic and Marine Weather-Forecast Models

9. Stewart, R. H., *Methods of Satellite Oceanography*, University Of California Press, Berkeley, 1985

10. Stoffelen, A. C. M., and Anderson, D. L. T., "The ECMWF Contribution to the Characterization, Interpretation, Calibration and Validation of ERS-1 Scatterometer Backscatter Measurements and Winds, and their use in Numerical Weather Prediction Models," European Space Agency Contract Report, ESA Contract Number 9097/90/NL/BI, 1995

11. Offiler, D., "The Calibration of ERS-1 Satellite Scatterometer Winds," *Journal of Atmospheric and Oceanic Technology*, v. 11, pp.1002-1017, 1994

12. Pierson, W. J., and Sylvester, W. B., "Concerns About the Operational Assimilation of Scatterometer Data," paper presented at the Workshop on the Operational Use of Scatterometer Measurements of the Ocean Surface Wind Field, Alexandria, Virginia, USA, 22-23 April, 1996.

13. Peters, C. A., Gerald, V. M., Woiceshyn, P. M., and Gemmill, W. H., "Operational Processing of ERS-1 Scatterometer Winds: A Documentation," National Oceanic and Atmospheric Administration Ocean Products Center Office Note, Contribution No. 96, 1994.

14. Offiler, D., "Final Study on the Development of ERS-1 Scatterometer Wind Retrieval Algorithms," European Space Agency Contract Report, ESA Contract Number EP/JVD/14.7/3320, 1987.

15. Ocean Model Support Program Reference Manual, Fleet Numerical Meteorology and Oceanography Center.

16. Nomura, A., "Global Sea Ice Concentration Data Set for Use With the ECMWF Re-Analysis System," European Center for Medium-Range Weather Forecasting Technical Report No. 76, March 1995.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center..2
   8725 John J. Kingman Road, Ste 0944
   Ft. Belvoir, Virginia 22060-6218

2. Library, Code 52...2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, California 93943-5101

3. Fleet Numerical Meteorology and Oceanography Center...................................2
   Code 70
   7 Grace Hopper Avenue
   Monterey, California 93943-5501

4. Marie Colton..1
   Office of Naval Research, Code 321RS
   800 North Quincy Street
   Arlington, Virginia 22217

5. Sandra Scrivener..1
   Naval Postgraduate School, Code AA/SS
   Monterey, California 93943-5101

6. LT(jg) Christy Martin...2
   245 Foremast Road
   Kingston, Tennessee 37763